Neurocomputing 122 (2013) 535–544

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Ensemble of online neural networks for non-stationary and imbalanced data streams

Adel Ghazikhani [a,*], Reza Monsefi [a], Hadi Sadoghi Yazdi [a,b]

[a] Computer Engineering Department, Ferdowsi University of Mashhad (FUM), Iran
[b] Center of Excellence on Soft Computing and Intelligent Information Processing, Ferdowsi University of Mashhad (FUM), Iran

A R T I C L E I N F O

Article history:
Received 7 January 2012
Received in revised form
11 March 2013
Accepted 22 May 2013
Communicated by R. Capobianco Guido
Available online 6 June 2013

Keywords:
Data stream classification
Online ensemble
Concept drift
Imbalanced data
Cost-sensitive learning

A B S T R A C T

Concept drift (non-stationarity) and class imbalance are two important challenges for supervised classifiers. "Concept drift" (or non-stationarity) refers to changes in the underlying function being learnt, and class imbalance is a vast difference between the numbers of instances in different classes of data. Class imbalance is an obstacle for the efficiency of most classifiers. Research on classification of non-stationary and imbalanced data streams, mainly focuses on batch solutions, whereas online methods are more appropriate. Here, we propose an online ensemble of neural network (NN) classifiers. Ensemble models are the most frequent methods used for classifying non-stationary and imbalanced data streams. The main contribution is a two-layer approach for handling class imbalance and non-stationarity. In the first layer, cost-sensitive learning is embedded into the training phase of the NNs, and in the second layer a new method for weighting classifiers of the ensemble is proposed. The proposed method is evaluated on 3 synthetic and 8 real-world datasets. The results show statistically significant improvement compared to online ensemble methods with similar features.

1. Introduction

Data stream classification (DSC) is gaining increasing attention in pattern recognition and machine learning community. This is related to its numerous applications such as credit card operations, weather, sensor networks, fraud detection, network attacks, web data, etc. The main issue in DSC is "concept drift", which is changes in the underlying function generating the data [1]. This requires the classifier model to be updated occasionally.

Previous literature reported three categories of methods for DSC namely, instance selection, instance weighting and ensemble models. In the "instance selection" (or sliding window) category, the algorithms have two phases, which are drift detection and retraining the classification model. The classification model is updated whenever drift is detected. These algorithms maintain a buffer of instances (usually recent instances) to retrain the classifier. Klinkenberg and Joachims [2] proposed a sliding window with adaptive size for handling concept drift in SVM. Sun and Li [3] utilized different variations of sliding window scheme for financial distress prediction. As for the second category "instance weighting", an incrementing weight is assigned to every incoming instance to make newer instances more effective in the classification model. Martinez-Rego et al. [4] proposed an online one-layer neural network (NN) that utilizes a forgetting function for instance weighting. Pavlidis et al. [5] proposed a classifier based on the recursive least square adaptive filter that incorporates an adaptive forgetting factor for instance weighting. And finally, ensemble models are the most frequently used methods for DSC.

In the literature, ensemble models have been used with two different settings, i.e. batch and online. The difference between batch and online learning is in the number of instances used to train the classifier [6]. In online learning, the classifier is trained with a single instance whereas in batch learning, a group of instances is used. In batch ensemble models, the data stream is divided into chunks. In this type of ensemble, a learner is built with each incoming chunk and learners are weighted according to their error on the current chunk. Elwell and Polikar [7] proposed a dynamic ensemble model that handles different kinds of concept drifts. Abdulsalam et al. [8] proposed an ensemble model that considers time constraints more seriously. Masud et al. [9] proposed an ensemble model that handles concept drift and concept evolution concurrently.

Online ensemble models are another type of ensemble models used for DSC. An online ensemble is the one which is composed of online classifiers [10]. In this type of ensemble, the number of classifiers is usually fixed and the classifiers are commonly weighted with the Winnow method [11,12]. Rodriguez and Kuncheva [13] proposed an ensemble of three online classifiers

* Corresponding author. Tel.: +98 1511 880 5065; fax: +98 151 187 633 04.
E-mail addresses: a_ghazikhani@yahoo.com, Adel.Ghazikhani@stu-mail.um.ac.ir (A. Ghazikhani), monsefi@um.ac.ir (R. Monsefi), sadoghi@um.ac.ir (H. Sadoghi Yazdi).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.05.003

namely, nearest neighbor, online perceptron and online linear discriminant for DSC. Kotsiantis et al. [14] proposed an ensemble of online perceptron, naïve bayes and 1-nearest neighbor. Abdel-hamid et al. [15] proposed an online ensemble in which the adaptation to concept drift is performed in three levels. Pocock et al. [16] proposed an online boosting method that uses classifier impact on ensemble performance to handle drift. Minku and Yao [17] proposed an online ensemble which incorporates diversity for handling drift.

In the literature, different classifiers have been proposed that cope with a combination of "concept drift" and other challenges [18]. One of these challenges is class imbalance. In imbalanced data, the number of data in different classes differs vastly. In two-class imbalanced data, the class with less data is called the minority, and the other is called the majority class. Imbalanced data cause uneven classification results in different classes. Class imbalance has been studied in various paradigms thoroughly [19–21]. Previous literature has made little attention to imbalanced data stream classification (IDSC). Algorithms for IDSC are composed of two main parts for handling concept drift and class imbalance. Gao et al. [22] utilized ensemble model for handling concept drift and oversampling for dealing with class imbalance. Chen and He [23] use the same strategy as Gao with some improvements in both parts. Ditzler and Polikar [24] proposed an algorithm that again utilizes an ensemble model for handling concept drift, but to handle class imbalance a modification in the ensemble learning scheme is proposed. Most of the previous methods on IDSC use batch solutions for training the classifiers. In these methods, the classifier is trained using a batch of data. Unlike batch processing, online processing has a per-instance view on the data stream [6]. In online learning, the learner is updated when a new instance arrives. This type of learning is more appropriate for IDSC.

Building on previous studies on IDSC, we proposed an online ensemble of NN classifiers for non-stationary and imbalanced data streams. The main contribution is a two layer approach for handling class imbalance and non-stationarity. The first layer handles class imbalance in each classifier of the ensemble with a cost-sensitive approach embedded in the training phase of the NNs. The second layer handles both class imbalance and non-stationarity at the ensemble level with a new classifier weighting method.

2. The proposed method

An online ensemble of classifiers is proposed for IDSC (Fig. 1).

The proposed method is constructed of two layers (Fig. 1). In the first layer, a cost-sensitive NN is proposed that is suitable for handling class imbalance and is the only type of learner in the ensemble. In the second layer, a new mechanism for weighting classifiers of an online ensemble is proposed. The proposed weighting method is based on the Winnow method. Winnow is a method for weighting classifiers in online ensembles [11]. The proposed weighting method handles "concept drift" (non-stationarity) and class imbalance concurrently.

The outputs of the classifiers are aggregated together with the majority vote combination scheme. Similar to previous research on online ensembles, the number of classifiers is fixed [11,12,14,15]. Another issue that should be considered in ensemble learning is diversity [25]. Diversity is the difference between the outputs of the individual learners in the ensemble. Here, we used different initial weights for generating diversity. Different initial weights is a method for generating diversity in an ensemble of NNs [26].

In the following sections the proposed cost-sensitive NN and ensemble weighting method are detailed.

2.1. Cost-sensitive neural network

Cost-sensitive learning is a paradigm in which learning of different classes requires different costs to be taken in account [27]. An example is medical diagnosis in which the cost of misclassifying data from the ill patient class is more than the healthy patient class. Cost-sensitive learning is one of the approaches used for handling class imbalance [28]. In this method, the misclassification cost of different classes is adjusted to have a more balanced classification result.

In this research, cost-sensitive NNs are proposed as the learners of the online ensemble model. The proposed cost-sensitive NN is based on Fontenla-Romero's proposed NN [29]. The architecture of this NN is shown in Fig. 2

The objective function of this one layer NN is

$$MSEB_j = \sum_{s=1}^{S} \left(f_j'(\bar{d}_{js}) \left(f_j^{-1}(d_{js}) - \sum_{i=0}^{I} w_{ji} x_{is} \right) \right)^2, \quad \forall j = 1, \ldots, J \tag{1}$$

In Eq. (1), S is the number of samples, J is the number of activation functions, f_j is the activation function of the jth neuron, $\bar{d}_{js} = f_j^{-1}(d_{js})$, d_{js} is the desired output, w_{ji} is the weight connecting the jth neuron to the ith feature of the input sample and x_{is} is the ith feature of the sample. This objective function was modified to include the cost-sensitive feature required to handle class imbalance. The new objective function is

$$MSEB_j = \sum_{s=1}^{S} c_j(s) \left(f_j'(\bar{d}_{js}) \left(f_j^{-1}(d_{js}) - \sum_{i=0}^{I} w_{ji} x_{is} \right) \right)^2, \quad \forall j = 1, \ldots, J \tag{2}$$

In Eq. (2), $c_j(s)$ is the function specifying the misclassification cost. For each sample, $c_j(s)$ is determined based on the observed output of the NN and the true label. The $c_j(s)$ function represents the cost matrix in cost-sensitive learning. The cost

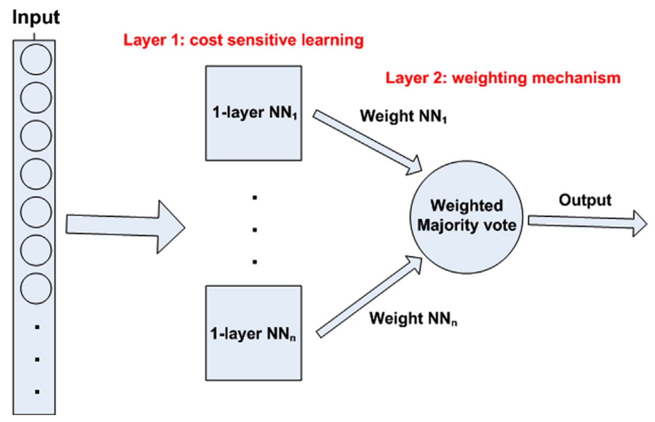

Fig. 1. The proposed method.

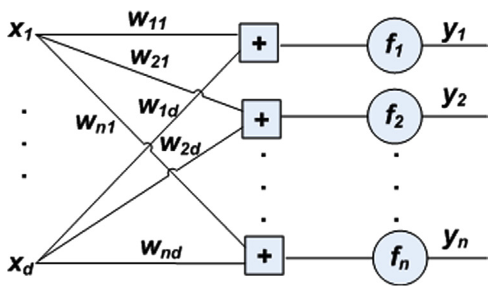

Fig. 2. Architecture of one-layer feedforward NN [29].

matrix is shown below

$$\text{Fixed Cost Matrix} = \begin{pmatrix} 1 & c_1 \\ c_2 & 1 \end{pmatrix} \quad (3)$$

In Eqs. (3), c1 and c2 are the misclassification costs of the positive and negative class (minor and major) respectively.

2.2. Winnow-based classifier weighting

Winnow is a method for weighting classifier votes in online ensemble models [12]. In this method, classifier weights are changed based on their predictions. The weight is increased when a classifier has correct prediction and decreased otherwise. Therefore, Winnow is a suitable for handling concept drift in online ensemble models for DSC. In this research, we proposed a new Winnow-based weighting method appropriate for non-stationary and imbalanced data streams. The pseudo code of the proposed algorithm is shown below

Algorithm. Winnow-based weighting for imbalanced data streams

```
1: Initialize weights $wmin_1,…,wmin_n$ and $wmaj_1,…,wmaj_n$,
 s.t. $\sum_{i=1}^{n} wmin_i = 1, \sum_{i=1}^{n} wmaj_i = 1$
   /*wmin is minority class weight, wmaj is majority class
   weight and n is number of classifiers*/
2: Initialize $\alpha, \beta$, and $\gamma$ as promotion parameter, demotion
   parameter and imbalance parameter s.t.
   $\alpha > 1, 0 < \beta < 1$ and $\gamma > 1$
3: for $t$=1,2,…
4: {
5: Get the current sample $(x_t,y_t)$
6: Get the prediction of each classifier $(p_1^t,…,p_n^t)$
  (1:minority class, 2:majority class) to $(x_t,y_t)$
7: Compute the weighted majority vote of the ensemble
   $\tilde{y}_t = \arg\max_j(\sum_{j=1}^{n}(p_j^t = 1) \times (\gamma \times wmin_j + wmaj_j)$
        $+ (p_j^t = 2) \times (wmin_j + wmaj_j))$
8: if $\tilde{y}_t \neq y_t$ then
9: {
10: for $j$=1:$n$
11: {
12: if $p_j^t = y_t$ then
13: {
14: if $y_t = 1$ then
15: $wmin_j^{t+1} = wmin_j^t \times \alpha$
16: else if $y_t = 2$ then
17: $wmaj_j^{t+1} = wmaj_j^t \times \alpha$
18: }
19: else
20: {
21: if $y_t = 1$ then
22: $wmin_j^{t+1} = wmin_j^t \times \beta$
23: else if $y_t = 2$ then
24: $wmaj_j^{t+1} = wmaj_j^t \times \beta$
25: }
26: }//end for
27: $wmin_j = \frac{wmin_j^{t+1}}{\sum_{j=1}^{n} wmin_j^{t+1}}, wmaj_j = \frac{wmaj_j^{t+1}}{\sum_{j=1}^{n} wmaj_j^{t+1}}$//normalize weights
28:}//end if
29:}//end for
```

The main innovation is considering two separate weights for the minority and majority classes (*wmin and wmaj*). In other words, the performance of each classifier on each class of data is considered separately. This feature enforces a new majority voting scheme for combining the classifiers

$$\tilde{y}_t = \arg\max_j\left(\sum_{j=1}^{n}(p_j^t = 1) \times (\gamma \times wmin_j + wmaj_j) + (p_j^t = 2) \times (wmin_j + wmaj_j)\right) \quad (4)$$

As shown in Eq. (4), there are separate weights for each class of data (*wmin and wmaj*). In the majority voting scheme, there are two different terms specifying the weight of each classifier's vote. These weights are specified by the prediction result in the minority or majority class ($p_j^t = 1$ or $p_j^t = 2$). As shown in Eq. (4), the weight of a classifier vote in the minority class is enhanced by the γparameter. This type of weighting results in better classification results in imbalanced data streams.

3. Experimental results

In this section experiments were conducted to evaluate the effectiveness of the proposed method. Experiments were performed on artificial and real-world two-class datasets. Since our research was on IDSC, the datasets were preprocessed to simulate high imbalance ratio. By doing so, instances were randomly removed from the minor class to simulate high imbalance ratio. This type of preprocessing has been used in previous research on IDSC [23]. In all the experiments, the imbalance ratio was set to 0.05. This preprocessing was repeated 10 times for each dataset and the obtained results were the average results of these 10 random samples.

The number of classifiers in the online ensemble was set to 5 and sigmoidal activation function was used in the NNs. In the cost-sensitive NN, the cost matrix is

$$\text{Cost matrix} = \begin{pmatrix} 1 & c_1 \\ c_2 & 1 \end{pmatrix}$$

In this cost matrix c_1 is 2 and c_2 is 1 (misclassification costs of the minority and majority class). In the Winnow-based weighting method, α, β were set randomly and γ, the imbalance parameter, was set to 5. Previous methods for IDSC mainly focus on batch solutions [22–24], whereas our proposed method is an online solution. Therefore we define four different algorithms to evaluate the effectiveness of the proposed method (Table 1).

The algorithms listed in Table 1 represent two main features that are the type of classifier and the weighting method. Each of these features has two states that are original method and proposed method. Combining these two features we come up with four algorithms. The proposed algorithm is Alg4. In Table 1, the original one-layer NN is Fontenla-Romero's proposed NN [29] and the original weighting method is Kuncheva's standard winnow method [12].

3.1. Evaluation

In this section, we present some issues related to the evaluation scheme.

Table 1
Compared algorithms

Algorithm	One layer NN	Weighting method
Alg1	Original	Original
Alg2	Proposed cost sensitive	Original
Alg3	Original	Proposed weighting
Alg4	Proposed cost sensitive	Proposed weighting

3.1.1. *Evaluation metrics for imbalanced data*

An important issue in classification problems is the evaluation metric. Usually, classification accuracy is used to evaluate the result of classifiers. This metric is not suitable for imbalanced datasets, because the classifier has imbalanced results in different classes of data. Therefore in class imbalance learning, other metrics that represent the accuracy in each class are used [28].

In this study, we use the geometric mean metric for evaluating the algorithms

$$\text{Geometric mean} = \sqrt{TPR \times TNR}$$
$$TPR = \frac{TP}{TP+FN}, TNR = \frac{TN}{TN+FP} \tag{5}$$

In Eq. (5) *TPR* and *TNR* are the true positive rate and the true negative rate respectively. *TPR* is the number of correctly predicted instances from the positive class divided by the whole number of positive instances and *TNR* is the number of correctly predicted instances from the negative class divided by the whole number of negative instances.

3.1.2. *Evaluation method*

Online classifiers could be evaluated in two settings. The first one is overall evaluation, in which the metric is evaluated on the whole dataset. The second one is incremental learning curve which is a two-dimensional curve whose horizontal axis corresponds to the number of instances that have been classified by time *t* and whose vertical axis corresponds to the metric value. This curve represents the effectiveness of the approach in every instant.

3.2. *Synthetic datasets*

In this section, we present experiments on artificial datasets. All of these datasets have been used in previous researches on DSC.

3.2.1. *SEA*

SEA is a frequently used synthetic dataset in previous stream mining researches [30]. This dataset contains three features with random values in [0, 10]. The label of the instances is specified using a threshold for the sum of the first two features. Concept drift is added to the dataset by adjusting the threshold periodically. This dataset simulates an abrupt change in the class concepts.

Similar to the original design of the SEA dataset, the whole data stream is categorized into 4 blocks, and for each of these blocks, the threshold is fixed and set to 8, 9, 7 and 9.5. In each block, 12,000 instances are generated. Examples in which the sum of two features is greater than the threshold, belong to the majority class, and others belong to the minority class. The results are detailed in Table 2 and Fig. 3.

As shown in Table 2 and Fig. 3, the proposed algorithm has better results compared to other algorithms. The results show that in this dataset the cost-sensitive element is more effective than the weighting method. The low results are due to the imbalance feature of the dataset.

Table 2
Results for SEA dataset

Algorithm	TPR	TNR	GM
Alg1	0.0033	0.9998	0.0534
Alg2	0.0042	0.9997	0.0625
Alg3	0.0187	0.9998	0.1363
Alg4 (proposed method)	0.0199	0.9997	0.1407

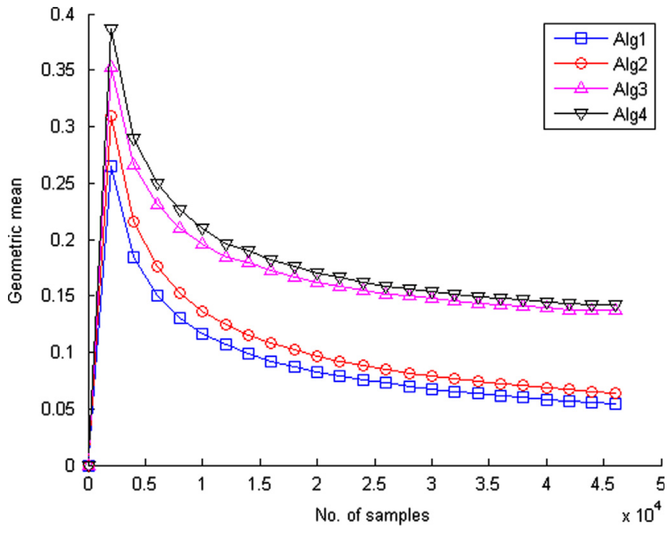

Fig. 3. Incremental learning curve for the SEA dataset.

Table 3
Feature values and classes of STAGGER dataset

Feature	First value	Second value	Third value
Size	Small	Medium	Large
Color	Red	Green	Blue
Shape	Square	Circular	Triangular
Classes			
1	size = small AND color = red		
2	(color = green OR shape = circular) OR (size = medium OR size = large)		

3.2.2. *STAGGER*

The STAGGER dataset is a popular artificial dataset used in previous stream mining research [31]. This dataset contains three categorical features, each having three distinct values. The dataset features and classes are detailed in Table 3.

The original STAGGER dataset has three concepts (classes), but here we reduced it to two concepts. To achieve this, the condition of the third concept was used for the second concept. In this dataset, the concepts are drifted by alternating the conditions. The experiment results are detailed in Table 4 and Fig. 4.

In Fig. 4, it can be observed that Alg2 has better results than Alg3. Therefore, unlike the SEA dataset, the weighting method is more effective than the cost-sensitive element. This implies that the performance of these two elements depends on the nature of the dataset.

3.2.3. *Moving hyperplane*

The moving hyperplane is a popular stream dataset that simulates gradual drift [32]. In this dataset, a hyperplane is constructed in *n* dimensions using coefficients $(\alpha_1, \alpha_2, ..., \alpha_n)$ as below

$$MH = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + ... + \alpha_n x_n \tag{6}$$

Each example in the dataset is created with each of the *n* features having a value in [0, 1]. The bias $\alpha_0$is calculated as

$$\alpha_0 = \frac{1}{2}\sum_{i=1}^{n} \alpha_i \tag{7}$$

The class labels are assigned using

$$y = \begin{cases} 1 & \sum_{i=1}^{n} \alpha_i x_i \geq \alpha_0 \\ 0 & \sum_{i=1}^{n} \alpha_i x_i < \alpha_0 \end{cases} \tag{8}$$

Table 4
Results for the STAGGER dataset

Algorithm	TPR	TNR	GM
Alg1	0.0008	0.9988	0.0122
Alg2	0.0037	0.9957	0.0516
Alg3	0.0010	0.9988	0.0139
Alg4 (proposed method)	0.0042	0.9959	0.0586

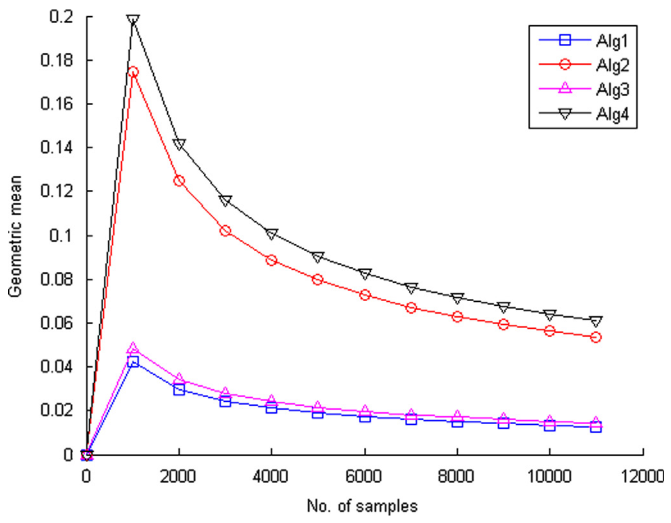

Fig. 4. Incremental learning curve for the STAGGER dataset.

Table 5
Results for the Moving hyperplane dataset.

Algorithm	TPR	TNR	GM
Alg1	0.0002	0.9999	0.0100
Alg2	0.0004	0.9996	0.0176
Alg3	0.0002	0.9999	0.0100
Alg4 (proposed method)	0.0004	0.9996	0.0176

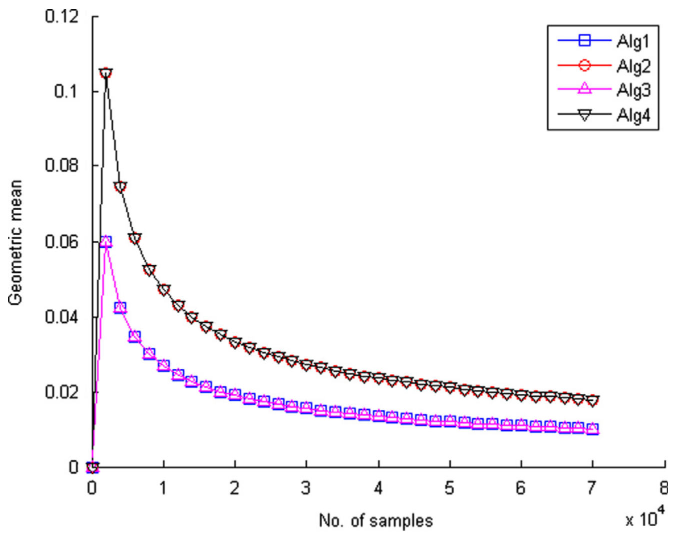

Fig. 5. Incremental learning curve for the hyperplane dataset.

The dataset was generated by rotating the hyperplane from 0 to 360, incrementing 5° in each step. At each step 1000 instances were generated. The experiment results are detailed in Table 5 and Fig. 5.

As shown in Table 5 and Fig. 5, Alg2 and Alg4 which both have the cost-sensitive element in common, have better results.

3.3. Real-world datasets

In this section, we present experiments on real-world datasets. All of these datasets have been used in previous researches on DSC.

3.3.1. Electricity market

The electricity market dataset is a real world data stream collected from the Australian New South Wales Electricity Market between the years 1996 and 1998 [33]. The original dataset contains 45,312 records, but since some instances have missed feature values, we only retain the records after May 11, 1997, which sums up to 27,549 records. The original dataset has eight features, but here, we only use the last four features, which are NSW electricity demand, the VIC Price, the VIC electricity demand, and the scheduled transfer between states. The experiment results are detailed in Table 6 and Fig. 6.

As shown in Table 6 and Fig. 6, Alg3 and Alg4, which both have the same weighting method, have better results.

3.3.2. PAKDD 2009 credit card

The PAKDD 2009 Credit Card dataset was collected from credit card operations at a major Brazilian retail chain during stable inflation condition through a 1-year period [34]. This dataset contains 50,000 data points each represented by 27 features related to a client credit card operation, such as age, sex, etc. The target class is the client's status that is good or bad. The experiment results are detailed in Table 7 and Fig. 7.

Table 6
Results for the Electricity Market dataset.

Algorithm	TPR	TNR	GM
Alg1	0.0176	0.9987	0.1283
Alg2	0.0176	0.9987	0.1283
Alg3	0.0242	0.9984	0.1518
Alg4 (proposed method)	0.0248	0.9984	0.1534

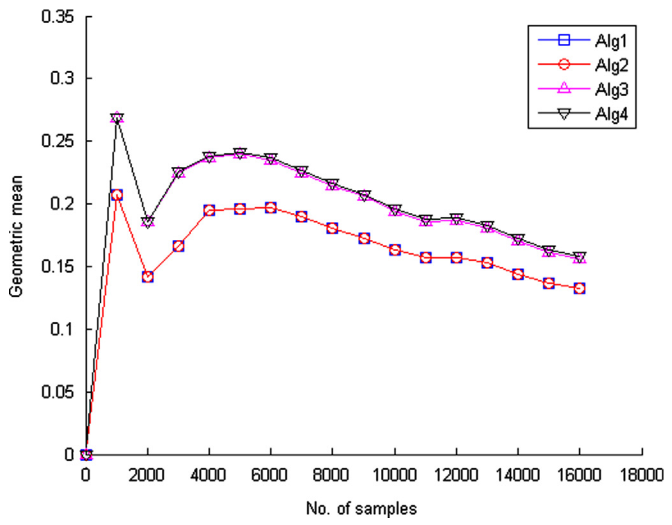

Fig. 6. Incremental learning curve for the Electricity Market dataset.

Table 7
Results for the Credit Card dataset.

Algorithm	TPR	TNR	GM
Alg1	0.0014	0.9991	0.0353
Alg2	0.0032	0.9975	0.0562
Alg3	0.0016	0.9989	0.0386
Alg4 (proposed method)	0.0044	0.9968	0.0649

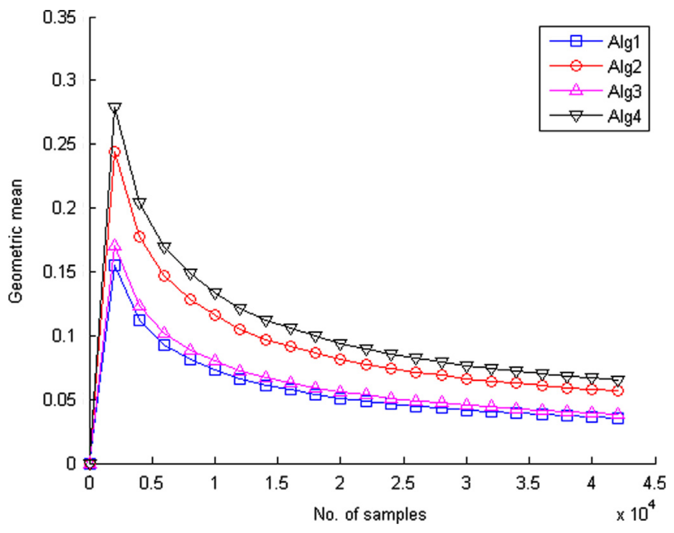

Fig. 7. Incremental learning curve for the Credit Card dataset.

Table 8
Results for the Weather dataset.

Algorithm	TPR	TNR	GM
Alg1	0.0015	0.9991	0.0318
Alg2	0.0020	0.9986	0.0361
Alg3	0.0034	0.9973	0.0563
Alg4 (proposed method)	0.0060	0.9939	0.0759

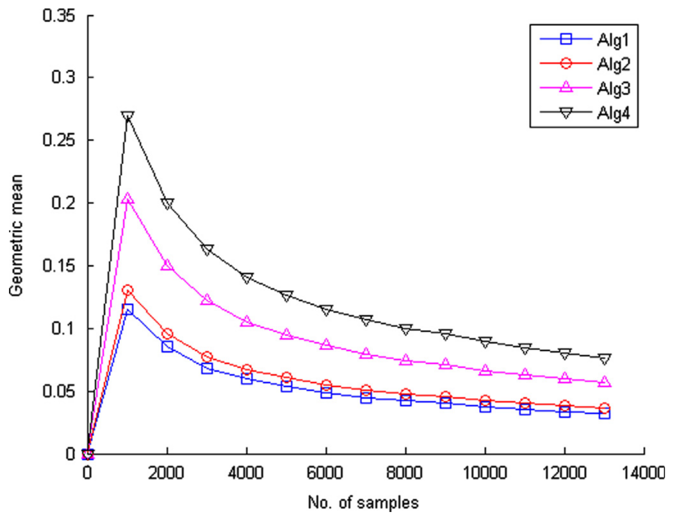

Fig. 8. Incremental learning curve for the Weather dataset.

3.3.3. Weather

The Weather dataset was created by the National Oceanic and Atmospheric Administration (NOAA), part of the United States Department of Commerce (USDC) [35]. This dataset was generated by compiling a database of weather measurements from over 7000 weather stations worldwide between 1949 and 1999. This dataset contains 18,159 data points, each represented by 8 features that are related to the weather condition (for example, temperature, dew point, etc.). The target class is the weather status that is ‘rain’ or ‘no rain’. The experiment results are detailed in Table 8 and Fig. 8.

Table 9
Results for the UCI dataset.

Metrics	TPR	TNR	GM
Dataset/Algorithms			
Haberman			
Alg1	0	0.9999	0
Alg2	0.0001	0.9998	0.0041
Alg3	0.0003	0.9998	0.0086
Alg4 (proposed method)	0.0004	0.9997	0.0136
Transfusion			
Alg1	0.0005	0.9999	0.0154
Alg2	0.0010	0.9996	0.0226
Alg3	0.0475	0.9900	0.2111
Alg4 (proposed method)	0.0828	0.9718	0.2764
Pima			
Alg1	0.0002	0.9998	0.0055
Alg2	0.0003	0.9995	0.0103
Alg3	0.0002	0.9997	0.0055
Alg4 (proposed method)	0.0004	0.9994	0.0135
Yeast			
Alg1	0.0055	0.9988	0.0733
Alg2	0.0260	0.9880	0.1595
Alg3	0.0047	0.9989	0.0669
Alg4 (proposed method)	0.0240	0.9889	0.1534
E. coli			
Alg1	0.0479	0.9981	0.2182
Alg2	0.0820	0.9728	0.2823
Alg3	0.1582	0.9982	0.3970
Alg4 (proposed method)	0.1903	0.9704	0.4295

3.4. UCI datasets

In this section, we report results on data streams generated from UCI datasets [36]. Yang et. al [37] proposed an iterative method to convert a normal dataset to a non-stationary data stream, using the Markov Chain concept. In this iterative method, a random attribute is selected in each iteration. If the attribute is nominal, each of its values is treated as a state in the Markov Chain. For each state, 2000 instances with the same attribute value are selected and added to the final data stream. On the other hand, if the attribute is numeric, the instances are sorted based on the selected attribute in ascending order and added to the final data stream. This procedure is repeated until a desired number of data, for example 50,000, are reached. In the current research, this method was used to convert some imbalanced datasets from the UCI repository to imbalanced data streams with concept drift.

As shown in Table 9 and Fig. 9, the proposed method had better results compared to other algorithms.

4. Discussion

4.1. Statistical significance

In this section the analysis of variance (ANOVA) test is used to verify the statistical significance of the empirical results [38]. ANOVA is a statistical test used for comparing the response of several algorithms to single or multiple factors. Similar to other statistical tests, ANOVA uses hypothesis testing to verify the

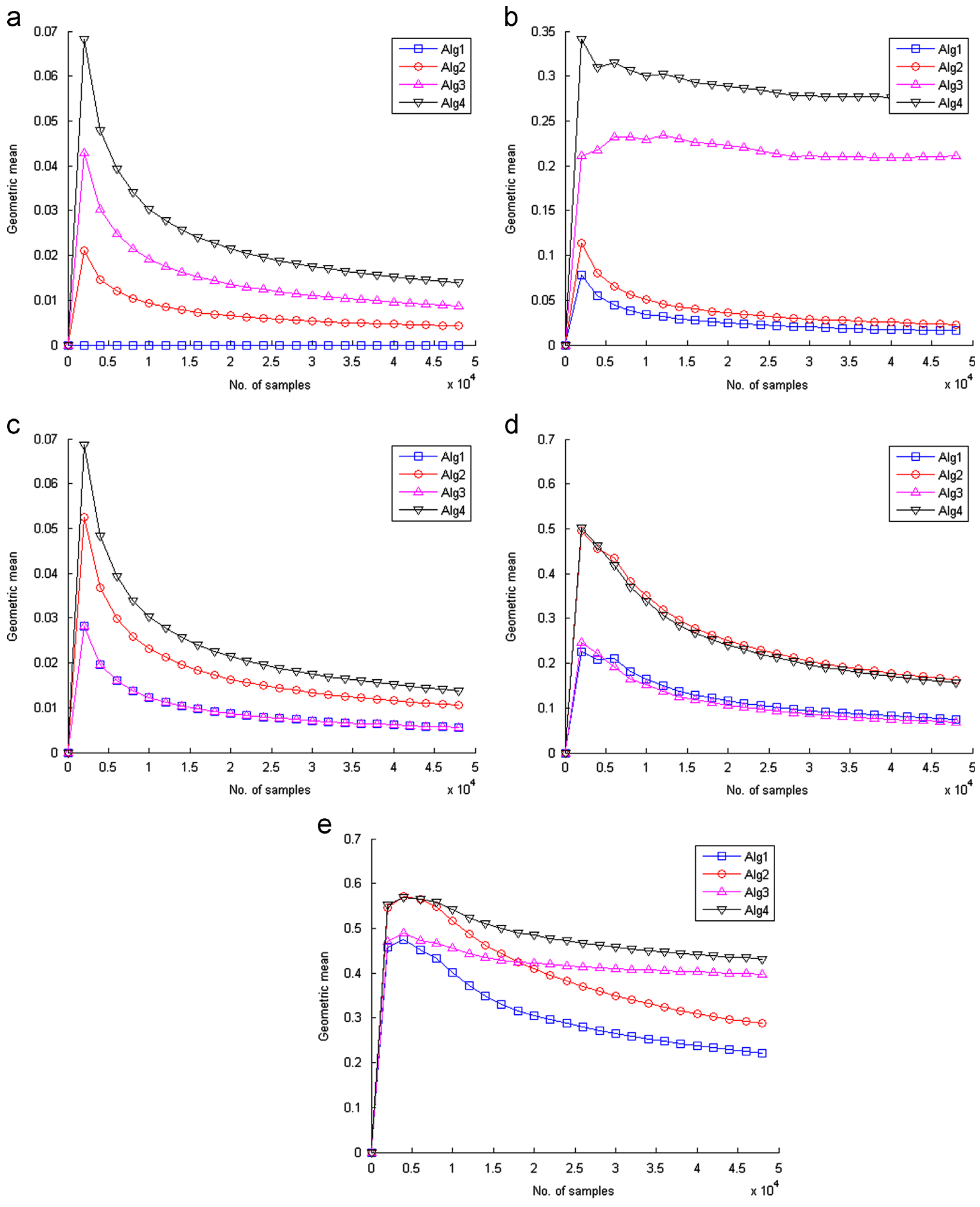

Fig. 9. Incremental learning curve (Geometric mean) for UCI datasets, (a) Haberman, (b) Transfusion, (c) Pima, (d) Yeast, (e) *E. coli*.

statistical significance. In hypothesis testing, a *null* hypothesis is specified which may be accepted or rejected by the statistical test. In ANOVA, the *null* hypothesis is when all the algorithms have similar average results. Here, the results of all the experiments are used to perform the ANOVA test. The significance factor (α) was set to 0.05. The ANOVA test rejected the *null* hypothesis for the algorithms. The box plot of the results is shown in Fig. 10.

Following the ANOVA test, the Turkey's Honestly Significance Difference (HSD) test was used to compare the results of every pair of algorithms. The result of the HSD test is detailed in Table 10. Alg4 has the best results. Alg4 has significantly better results than Alg1 and Alg2. Alg3 has significantly better results than Alg1.

4.2. *Parameter sensitivity*

In this section, we investigate the sensitivity of results to parameter values. The parameters of the proposed method are the cost matrix parameters namely, c_1 and c_2, and the parameters of ensemble learner weighting method which are α,β and γ. Several experiments were conducted to show the sensitivity of results to changes in parameter values. In each experiment, the value of a single parameter is changed and the rest of parameters have fixed values. For the cost matrix, the c_1 parameter is investigated. The SEA and Weather datasets were selected for the parameter sensitivity experiments. Fig. 11 shows the results.

The results imply that α and β parameters do not have much influence on the results. In other words, if α and β are adjusted according to the algorithm, increasing or decreasing one of them does not affect the results. On the other hand, increasing γ and c_1 parameters advance the results. These two parameters provide better accuracy for data in the minority class, resulting in a more balanced result.

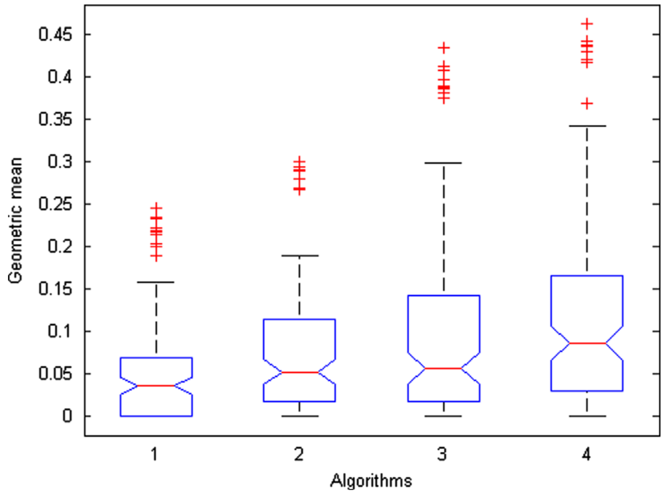

Fig. 10. ANOVA test result, 1:Alg1, 2:Alg2, 3:Alg3, 4:Alg4.

Table 10
Results of HSD test.

Algorithm	Average Geometric Mean	Significant from
Alg1	0.0530	Alg3,Alg4
Alg2	0.0756	Alg4
Alg3	0.0996	Alg1
Alg4 (proposed method)	0.1270	Alg1,Alg2

4.3. Sensitivity to imbalance ratio

Imbalanced datasets are characterized by the imbalance ratio of data in different classes. Imbalance ratio specifies the amount of difference between the numbers of instances in different classes. In the entire experiments, imbalance ratio was set to 0.05.

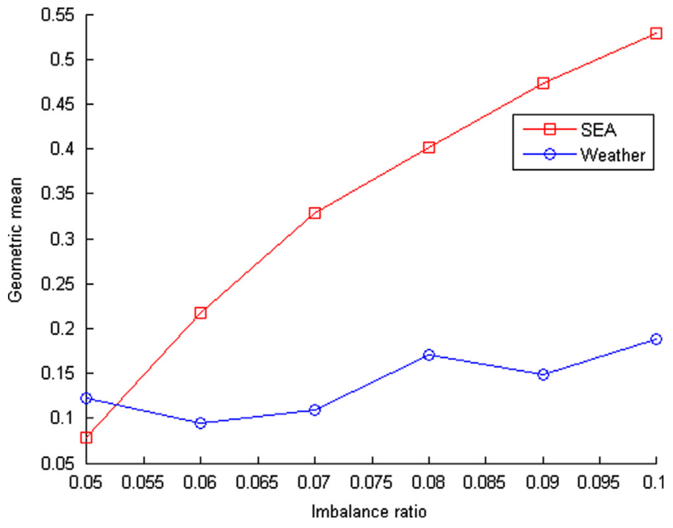

Fig. 12. Sensitivity to imbalance ratio for SEA and Weather dataset.

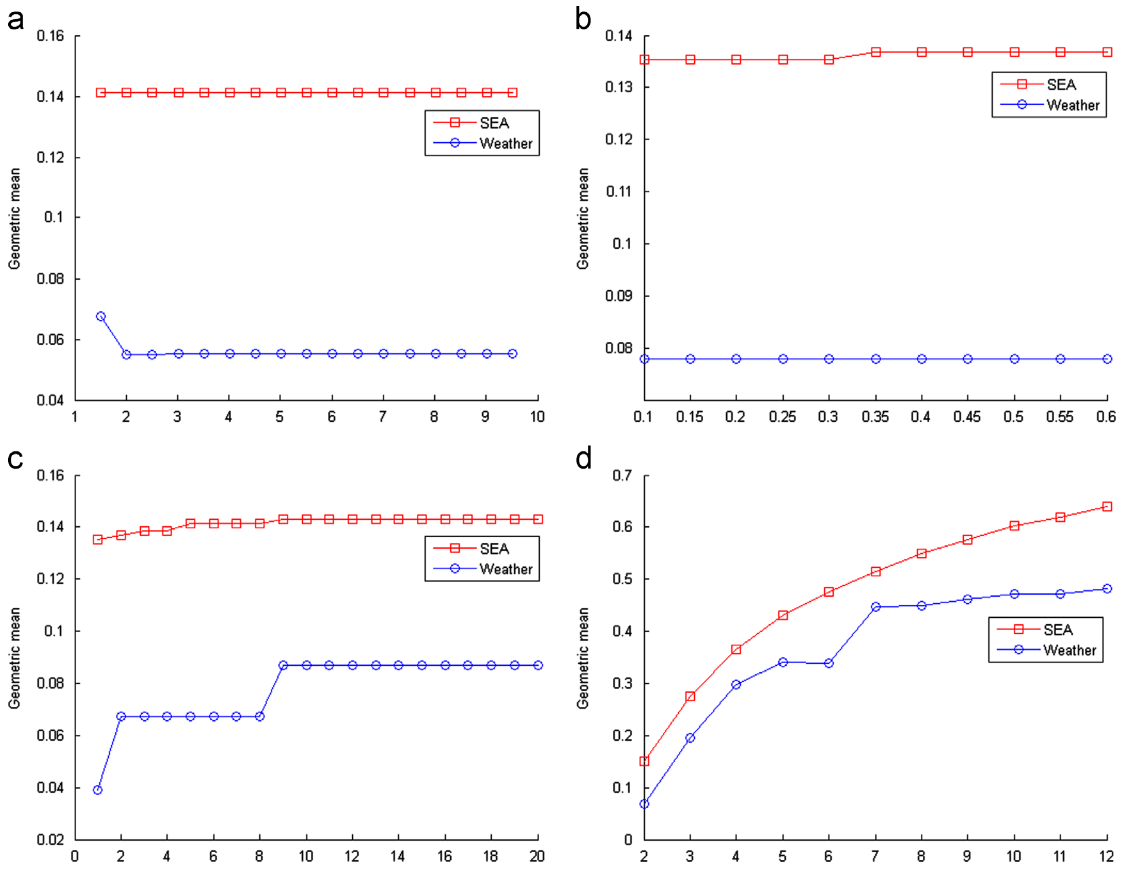

Fig. 11. Parameter sensitivity experiment results, (a) alpha, (b) beta (c) imbalance parameter (gamma) (d) cost (c_1).

Here, we conducted several experiments to witness the sensitivity of results to imbalance ratio. The experiments were performed on the SEA and Weather datasets. In each experiment, the imbalance ratio was changed between 0.05 and 0.1 and the proposed method was evaluated 5 times. The reported results are the average geometric mean for these 5 runs.

Fig. 12 shows the sensitivity of results to the imbalance ratio of the datasets. It could be concluded that increasing the imbalance ratio heightens the results. This result verifies that balanced datasets have balanced accuracy.

5. Conclusion

We have proposed an online ensemble of classifiers for non-stationary and imbalanced data streams. Imbalanced data stream classification (IDSC) embraces two significant challenges, which are non-stationarity and class imbalance. In the proposed online ensemble, the classifiers are Neural Networks (NN). The main contribution is a two-layer approach for handling class imbalance and non-stationarity. In the first layer, cost-sensitive NNs are proposed to handle class imbalance. In the cost-sensitive approach, more importance is assigned to errors in the minority class. This importance is implemented by weights embedded into the NN objective function. In the second layer, a novel method for weighting learners in the ensemble is proposed in which class imbalance and non-stationarity are handled simultaneously. In this method, separate weights are assigned to errors in the minority and majority classes and again errors in the minority class have more cost. The proposed method has been evaluated using the geometric mean metric, which is a metric for evaluating methods on imbalanced datasets [28]. The experimental results showed statistically significance of the proposed method. Statistical significance was verified using the ANOVA test followed by the HSD test with a significance factor (α) of 0.05.

The proposed method has five parameters namely, c_1, c_2, α, β and γ. The first two (c_1, c_2) are the parameters of the cost matrix and α, β and γ are the parameters of the proposed ensemble learner weighting method. The parameter sensitivity experiments demonstrate growth of α and β does not affect the results significantly but growth of γ and c_1 improves the results. The results show that the two main elements of the proposed method namely cost-sensitive NN and the weighting mechanism, have different behavior in each dataset.

Future research on this subject could have different directions which are adaptive cost-sensitive classifiers, an automated weighting method, an adaptive ensemble, and an ensemble with different types of classifiers.

References

[1] J. Gama, Knowledge Discovery from Data Streams, Chapman & Hall/CRC Press, 2010.
[2] R. Klinkenberg, T. Joachims, Detecting concept drift with support vector machines, Presented at the 17th International Conference on Machine Learning, San Mateo,CA, 2000.
[3] J. Sun, H. Li, Dynamic financial distress prediction using instance selection for the disposal of concept drift, Expert Syst. Appl. 38 (2011) 2566–2576.
[4] D. Martínez-Rego, et al., A robust incremental learning method for non-stationary environments, Neurocomputing 74 (2011) 1800–1808.
[5] N.G. Pavlidis, et al., Landa perceptron: an adaptive classifier for data streams, Pattern Recognition 44 (2011) 78–96.
[6] A. Tsymbal, The Problem of Concept Drift: Definitions and Related Work, Trinity College Dublin, Computer Science Department, Dublin, 2004.
[7] R. Elwell, R. Polikar, Incremental learning of concept drift in nonstationary environments, IEEE Trans. Neural Networks 22 (2011) 1517–1531.
[8] H. Abdulsalam, et al., Classification using streaming random forests, IEEE Trans. Knowl. Data Eng. 23 (2011) 22–36.
[9] M.M. Masud, et al., Classification and novel class detection in concept-drifting data streams under time constraints, IEEE Trans. Knowl. Data Eng. 23 (2011) 859–874.
[10] A. Fern, R. Givan, Online ensemble learning: an empirical study, Mach. Learn. 53 (2003) 71–109.
[11] N. Littlestone, Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm, Mach. Learn. 2 (1988) 285–318.
[12] L. Kuncheva, in: F. Roli, et al., (Eds.), Classifier Ensembles for Changing Environments Multiple Classifier Systems, vol. 3077, Springer, Berlin/Heidelberg, 2004, pp. 1–15.
[13] J.J. Rodriguez, L.I. Kuncheva, Combining online classification approaches for changing environments, Presented at the Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, Orlando, Florida, 2008.
[14] S. Kotsiantis, et al., A combinational incremental ensemble of classifiers as a technique for predicting students' performance in distance education, Knowledge-Based Syst. 23 (2010) 529–535.
[15] B. Abdelhamid, Incremental learning with multi-level adaptation, Neurocomputing 74 (2011) 1785–1799.
[16] A. Pocock, et al., Online Non-stationary Boosting Multiple Classifier Systems, in: N. El Gayar, et al., (Eds.), Springer, Berlin/Heidelberg, 2010, pp. 205–214.
[17] L. Minku, X. Yao, DDD: a new ensemble approach for dealing with concept drift, IEEE Trans. Knowl. Data Eng. vol. PP (2011) 1-1.
[18] X. Wu, et al., Learning from concept drifting data streams with unlabeled data, Neurocomputing 92 (2012) 145–155.
[19] H. Yu, et al., ACOSampling: an ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data, Neurocomputing 101 (2013) 309–318.
[20] S.-H. Oh, Error back-propagation algorithm for classification of imbalanced data, Neurocomputing 74 (2011) 1058–1061.
[21] C.-Y. Yang, et al., Margin calibration in SVM class-imbalanced learning, Neurocomputing 73 (2009) 397–411.
[22] J. Gao, et al., A General Framework for Mining Concept-Drifting Data Streams with Skewed Distributions, Presented at the SIAM, 2007.
[23] S. Chen, H. He, Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach, Evolving Syst. 2 (2010) 35–50.
[24] G. Ditzler, R. Polikar, An Ensemble Based Incremental Learning Framework for Concept Drift and Class Imbalance, Presented at the WCCI, 2010.
[25] L.I. Kuncheva, Combining Pattern Classifiers Methods and Algorithms, JOHN WILEY & SONS, 2004.
[26] G. Brown, et al., Diversity creation methods: a survey and categorisation,, Inf. Fusion 6 (2005) 5–20.
[27] C. Elkan, The foundations of cost-sensitive learning, Presented at the Proceedings of the 17th International Joint Conference on Artificial Intelligence, vol. 2, Seattle, WA, USA, 2001.
[28] H. He, E.A. Garcia, Learning from Imbalanced Data,, IEEE Trans. Knowl. Data Eng. 21 (2009) 1263–1284.
[29] O. Fontenla-Romero, et al., A new convex objective function for the supervised learning of single-layer neural networks, Pattern Recognition 43 (2010) 1984–1992.
[30] N.W. Street, Y..Kim, A streaming ensemble algorithm (SEA) for large-scale classification, Presented at the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001.
[31] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Mach. Learn. 23 (1996) 60–101.
[32] A. Narasimhamurthy, L.I. Kuncheva, A framework for generating data to simulate changing environments, Presented at the IASTED International Conference on Artificial Intelligence and Applications, 2007.
[33] M. Harries, Splice-2 Comparative Evaluation: Electricity Pricing, University of South Wales, 1999.
[34] Neurotech. 2009, PAKDD 2009 Data Mining Competition. Available from: ⟨http://sede.neurotech.com.br:443/PAKDD2009/⟩.
[35] NOAA, 2010, Weather Data. Available from: ⟨http://users.rowan.edu/~polikar/research/NSE/⟩.
[36] A. Asuncion, D. Newman, UCI Repository of Machine Learning Database. Available from: ⟨http://www.ics.uci.edu/~mlearn/MLRepository.html⟩.
[37] Y. Yang, et al., Mining in anticipation for concept change: proactive-reactive prediction in data streams, Data Mining Knowl. Discovery 13 (2006) 261–289.
[38] E. Alpaydın, Introduction to Machine Learning, Second Edition, The MIT Press, 2010.

Adel Ghazikhani was born in Mashhad, Iran, in 1981. He received his B.S. and M.S. degrees in computer engineering from Ferdowsi University of Mashhad (FUM) in 2004 and Azad University of Mashhad in 2008 respectively. Currently he is a Ph.D. candidate at FUM in computer engineering. His research interests are online learning, supervised learning and machine vision.

Reza Monsefi was born in Ahwaz, Iran, in 1956. He received his B.S. in electrical engineering from Manchester University in 1978, and received his M.S. and Ph.D. in control engineering from Salford University 1981 and 1983 respectively. Currently he is an Associate Professor at Ferdowsi University of Mashhad. His research interests are machine learning and pattern recognition.

Hadi Sadoghi Yazdi was born in Sabzevar, Iran, in 1971. He received his B.S. in Electrical Engineering from Ferdowsi University of Mashhad (FUM) 1994, and received his M.S. and Ph.D. in Electrical Engineering from Tarbiat Modares University Tehran in 1996 and 2005, respectively. Currently he is an Associate Professor at FUM. Dr. Sadoghi Yazdi has received several awards including Outstanding Faculty Award from Iran Ministry of Science, Research and Technology (MSRT) in 2009, 2010, 2011 and Best System Design Award from Ferdowsi Festival in 2007. His research interests are pattern recognition, machine learning, machine vision, signal processing, data mining and optimization.