Big Data and Information Analytics
©American Institute of Mathematical Sciences
Volume **2**, Number **1**, January **2017**
doi:10.3934/bdia.2017005
pp. **1–21**

SELECTIVE FURTHER LEARNING OF HYBRID ENSEMBLE FOR CLASS IMBALANCED INCREMENT LEARNING

Minlong Lin* and Ke Tang*

School of Computer Science and Technology
University of Science and Technology of China
HeFei, AnHui 230027, China
Springfield, MO 65801-2604, USA

Abstract. Incremental learning has been investigated by many researchers. However, only few works have considered the situation where class imbalance occurs. In this paper, class imbalanced incremental learning was investigated and an ensemble-based method, named Selective Further Learning (SFL) was proposed. In SFL, a hybrid ensemble of Naive Bayes (NB) and Multilayer Perceptrons (MLPs) were employed. For the ensemble of MLPs, parts of the MLPs were selected to learning from the new data set. Negative Correlation Learning (NCL) with Dynamic Sampling (DyS) for handling class imbalance was used as the basic training method. Besides, as an additive model, Naive Bayes was employed as an individual of the ensemble to learn the data sets incrementally. A group of weights (with the number of the classes as the length) are updated for every individual of the ensemble to indicate the confidence of the individual learning about the classes. The ensemble combines all of the individuals by weighted average according to the weights. Experiments on 3 synthetic data sets and 10 real world data sets showed that SFL was able to handle class imbalance incremental learning and outperform a recently related approach.

1. **Introduction.** Normal machine learning problems require learning model to learn information from all the achieved data and all the data are stored. However, in practice, the data are usually updated all the time and new information is necessary to be learned from the new data [19]. It is usually time consuming to learn new information with accessing to the previous data and storing the learned data is also expensive. In this situation, the learning model is required to have the ability of learning new information from new data and preserving the previously learned information without accessing the previous data. This learning model is called incremental learning [8],[21].

In incremental learning, the whole data set is not available in a lump. In another word, we can only get a part of the whole data set every time. We suppose that the whole data set S is divided into T subsets, i.e., $S_1, S_2, , S_T$. The rules (e.g., classification boundaries in classification problems) of S and S_t are denoted as R and R_t respectively. The aim of the learning model is to learn R by learning R_t from S_t respectively. The main difficulty is that the previously learned rules may be forgotten when the model learns new rules from new data subsets, especially

2010 *Mathematics Subject Classification.* Primary: 58F15, 58F17; Secondary: 53C35.
Key words and phrases. Incremental learning, class imbalance, Naive Bayes, Multilayer Perceptrons, Dynamic Sampling.

when the rules of different data subsets are different. This phenomenon was called catastrophic forgetting. If $R_1 = R_2 = ... = R_T$, the learning model can learn R_1 form S_1 and R_1 will not be forgotten when new data subsets are learned. In this case, incremental learning is not real challenging. However, in practice, R_t are usually different between different data subsets, so the catastrophic forgetting may happen.

In our assumption, even though the rules are different between different data subsets, the target rules (i.e., R) are not changed. This phenomenon was also called virtual concept drift [28] and it is different from real concept drift, in which the target concept is changed when new data subsets are available. Virtual concept drift was called sampling shift in [22] and it will be referred to in this paper. There are some additive models that can be easily adopted to learn incrementally when sampling shift occurs. For example, in Bayes Decision Theory, the rules can be represented by some parameters and the parameters of the whole data set can be combined by those of all the data subsets. In this way, the models can learn the data subsets respectively to form a same learner of learning the whole data set. However, these kinds of methods often require assumptions about the data distribution and the decision boundaries are always simple. Neural networks have strong abilities to learn complex classification boundary. Unfortunately, they are not additive. By training with new data subsets, the model tends to perform well on the new data subsets but poorly on the previous ones [8]. In other words, the model forgets the previously learned rules. Therefore, it is a challenge to employ neural networks to learn incrementally in this situation.

To exploit neural networks for incremental learning, some ensemble based approaches have been proposed. In our previous work, i.e., Selective Negative Correlation Learning (SNCL) [26], selective ensemble method was employed to pre-vent the model from forgetting previously learned information. There are also other ensemble based methods for incremental learning, such as Fixed Size Negative Correlation Learning (FSNCL), Growing Negative Correlation Learning (GNCL) [15], and Learn++ methods [21], [17], [5]. In SNCL and FSNCL (size-fixed methods), the model was able to learn new information from new data subsets with the size of the model fixed. How-ever, the ability of preserving previously learned information was not as good as Learn++ methods and GNCL (size-grown methods), in which the sizes of the models grown larger as new data subsets were learned. Since the new data subsets always become available all the time in practice, the sizes of the models will become too large in Learn++ methods and GNCL. Therefore, it is worthy to design a method with the benefits of both size-fixed methods and size-grown methods.

Besides sampling shift, there is another issue in incremental learning, i.e., class imbalance. In normal learning model, class imbalance problem has been studied by many researchers and there are plenty of literatures addressing class imbalance problems [11], [4], [9]. Class imbalance problems may also occur in incremental learning and this kind of issue has also been investigated [5], [6]. There are mainly two cases for class imbalance incremental learning:
(1)If the class distribution of the whole data set S is imbalanced, the class distribution of data subset S_t will usually be imbalanced. Furthermore, it will be common that samples of the minority classes may be lost in some data subsets.

(2)Even though the class distribution of S is balanced, S_t may also be class imbalanced. In typical case, all of the partial sets S_t are class imbalanced but the combined data set S is class balanced.

In this paper, we focus on class imbalance cases, in which sampling shift also occurs. Specifically, when sampling shift occurs, new classes may come up in the new data subset and some previous classes may be lost in the new data subset. When class distribution of the whole data set is imbalanced, this phenomenon will be more likely to happen to the minority classes. This is also the main issue in this paper.

The rest of the paper is organized as follows. In Section II, we will briefly review some existing methods for incremental learning. Our methods, i.e., Selective Further Learning (SFL) will be described in Section III. Then in Section IV, the experimental studies will be presented. Finally, we will conclude this paper and discuss the future work in Section V.

2. **Related work.** Some neural network based methods, such as the Adaptive Resonance Theory modules map (ARTMAP) [3] , [23], [29], [2] and Evolving Fuzzy Neural Network (EFuNN) [12] have been proposed for incremental learning. Both ARTMAP and EFuNN can learn new rules by changing the architecture of the models, such as self-organize new clusters (in ARTMAP) and create new neurons (in EFuNN) when new data are sufficiently different from previous ones. However, it is usually a non-trivial task to estimate the difference between new data and previous ones. Moreover, both of them are very sensitive to the parameters of the algorithms.

Researches have shown the good performance of ART-MAP and EFuNN in incremental learning. However, their abili-ties of learning incrementally in class imbalance situation have not been well investigated. In [5], where Learn++.UDNC was proposed for class imbalance incremental learning, fuzzy ARTMAP [2] was presented with poor performance when class imbalance occurs. Learn++.UDNC is an ensemble based method. It is one of the Learn++ series methods [21], [20] which were based on AdaBoost [7]. Besides Learn++.UDNC, many versions of Learn++ methods have been proposed, such as Learn++.MT [16], Learn++.MT2 [16], Learn++.NC [18] and Learn++.SMOTE [6]. In these versions, Learn++.MT and Learn++.NC was proposed for handling the problem of out-voting when learning new classes. Learn++.MT2 was pro-posed for handling the imbalance of examples between data subsets. These versions did not consider class imbalance situations. Class imbalance in incremental learning was addressed only in Learn++. UDNC and Learn++.SMOTE. In Learn++.UDNC, it was assumed that no real concept drift will happen, while in Learn++.SMOTE, real concept drift in class imbalanced data was investigated. Therefore, the former matches the issue in this paper but the later dose not.

Besides Learn++, another type of ensemble based methods, i.e., methods based on Negative Correlation Learning (NCL) [14], have also been proposed for incremental learning [26], [15]. NCL is a method to construct neural networks ensemble. It is capable of improving the generalization performance of the ensemble by decreasing the error of every neural network and increasing the diversities between neural networks simultaneously. In [15], tow NCL-based methods, i.e., FSNCL and GNCL were proposed. In FSNCL, the size of the ensemble is fixed and all of the neural networks are trained when new data subsets become available. In GNCL, the size of the ensemble grows as the data sets are incrementally learned and only

new added neural networks are trained when new data subsets become available. In our previous work [26], SNCL was proposed. In SNCL, new neural networks are added and trained when new data subsets become available and then a pruning method was employed to prune the ensemble to make the size of the ensemble fixed. Comparing to Learn++ methods and GNCL, FSNCL and SNCL can make the size of the ensemble fixed as more and more data sets come up while their abilities of preserving previously learned information are poorer than Learn++ and GNCL.

There are also some other methods with ability of incremental learning. Self-Organizing Neural Grove (SONG) [10] is an ensemble based method with Self-Generating Neural Net-works (SGNNs) [27] as the individual learners. Incremental Backpropagation Learning Networks (IBPLN) [8] employed neural networks for incremental learning by making the weights of the neural network bounded and adding new nodes. However, they did not consider the class imbalance in incremental learning.

SFL(M, *MLPs-Training*, *NB-Training*, *Select*)
  Repeat the following steps as data subsets come up:
  if first data subset S_1 comes up
    NB =Call *NB-Training* to get a model of Naive Bayes with S_1;
    $EnsMLP$ =Call *MLPs-Training* to get an ensemble of M MLPs with S_1;
    $Ens = EnsMLP \cup NB$;
    $nl = nc$ =the number of classes in S_1
    for every ith individual in Ens
      Get a $nc \times nc$ matrix $\mathbf{N}^{(i)}$ on S_1 for calculating R_{ik} and P_{ik} by (3) and (4);
      Update μ_i by (5) and w_i by (6);
    end for
  else if new data subset S_t comes up
    Ens_{sel} =Call *Select* to select $\lfloor M/2 \rfloor$ MLPs from $EnsMLP$;
    Call *NB-Training* to update NB with S_t;
    $Ens_{sel} = Ens_{sel} \cup NB$;
    $Ens_{res} = EnsMLP - Ens_{sel}$;
    Call *MLPs-Training* to train the MLPs of Ens_{sel} with S_t;
    n_t = the number of classes in S_t;
    n_c = the number of classes in $\cup_{i=1,\ldots,t} S_i$;
    for every ith MLPs in Ens_{sel};
      $\mu_i = n_t/n_c$;
      Initialize $\mathbf{N}^{(i)} = \mathbf{0}_{nc \times nc}$;
    end for
    $Ens = Ens_{sel} \cup Ens_{res}$;
    for every ith individual in Ens;
      Get a $nc \times nc$ matrix $\mathbf{N}_t^{(i)}$ on S_t;
      Update $\mathbf{N}^{(i)}$: $\mathbf{N}^{(i)} = \mathbf{N}^{(i)} + \mathbf{N}_t^{(i)}$;
      Update w_i by (6);
    end for
  end if
  After the above steps, we get Ens as the final model;
end *SFL*

FIGURE 1. The Pseudo-Code For SFL

3. **Our method.**

3.1. **Framework.** In this paper, class imbalance is considered in incremental learning. In the existing work, Learn++.UDNC [5] was pro-posed for addressing this

issue and it has been shown more effective than other incremental learning methods which did not consider class imbalance situation. However, as a size-grown method, the size of the ensemble in Learn++.UDNC increases all the way as new data sets become available. The size of the ensemble may become too large. In our method, ensemble based method is also considered and at the same time, we aim at controlling the size of ensemble at an acceptable level.

In our previous work, i.e., SNCL [26], selective ensemble was used to keep the size of the ensemble fixed. When new data subset comes up, it is used to train the copy of the previous ensemble. The two ensembles are combined and half of the individuals in the ensemble are pruned to keep the size of the ensemble fixed. However, in this model, previous information loss may easily occur due to the pruning process based on the latest data subset. The ensemble will be biased to the latest data subset. Furthermore, if the rules of the latest data subset is quite different from that of the previous data subset, i.e., high sampling shift occurs, all of the individuals of the previous ensemble might be pruned. On the other hand, since SNCL was designed without considering class imbalance situation, it might not be good at handling class imbalance incremental problems.

To overcome the above drawbacks, we propose a new en-semble based approach for incremental learning, i.e., Selective Further Learning (SFL). In SFL, a hybrid ensemble with two kinds of base classifiers was used. First of all, a group of Multi-Layer Perceptrons (MLPs) are used. When new data subsets become available, half of the MLPs in the current ensemble are selected to be trained with the new data subsets. After training, the selected MLPs are laid back to the ensemble. No pruning process will be executed so that the risk of previous information forgetting is reduced. At the same time, as an additive model, Naive Bayes (NB) is used as a component of the ensemble to incrementally learn from new data subsets. In this way, the strong incremental learning ability of NB will help the ensemble to preserve the previous information if high sampling shift occurs.

In addition, a group of weights (namely impact weights) are constructed for every individual (including MLPs and NB). The weights and the outputs of the ith individual are denoted as $\{w_{ik}|k = 1, 2, ..., C\}$ and $\{o_{ik}k = 1, 2, ..., C\}$, respectively, where C is the number of the classes. The impact weight w_{ik} is designed to indicate the confidence of the output produced by the ith individual on class k. At the testing stage, for an example, the output of the ensemble is calculated by the weighted average of all individuals:

$$y_k = \sum_{i=1}^{M} w_{ik} o_{ik} / \sum_{i=1}^{M} w_{ik}, k = 1, 2, ..., C, \tag{1}$$

where y_k is the kth output of the ensemble and indicates the probability that the example belonging to class k, M is the number of the individuals in the ensemble. Equation (1) is used only at the testing stage. At the training stage, the output of the ensemble is calculated by the arithmetical average of the individuals.

w_{ik} is initialized as 0 at the initial stage and updated during learning every new data subset. When updating w_{ik}, two issues should be considered.

On one hand, the grade that the ith individual learn about the kth class is considered, i.e., the recall of the ith individual on class k and the precision of the ith individual on class k. w_{ik} should be high when both the recall and precision are high. To this end, the definition of F-measure for multi-class [24] is introduced:

$$F_{ik} = \frac{2R_{ik}P_{ik}}{R_{ik} + P_{ik}} \tag{2}$$

where F_{ik}, R_{ik} and P_{ik} is the F-measure, recall and the precision of the ith individual on class k, respectively. According to [24], R_{ik} and P_{ik} are defined as

$$R_{ik} = \frac{N_{kk}^{(i)}}{\sum_{m=1}^{C} N_{km}^{(i)}} \tag{3}$$

and

$$P_{ik} = \frac{N_{kk}^{(i)}}{\sum_{m=1}^{C} N_{mk}^{(i)}} \tag{4}$$

where $N_{km}^{(i)}$ is the number of the examples of class k that were classified as class m by the ith individual.

On the other hand, since MLPs could be easily biased to the latest data subset, if some classes in the previous data subsets do not come up in the new data subset, the output of the MLPs that are selected to be trained with the new data subset should be suspectable. Therefore, a coefficient μ_i is defined for every individual i to degrade the impact weights:

$$\mu_i = \frac{nt}{nc}, \tag{5}$$

where nt is the number of classes that are contained in the new data subset, nc is the number of classes in all the coming up data subsets.

By considering both of the above issues, w_{ik} is updated as:

$$w_{ik} = F_{ik}\mu_i. \tag{6}$$

For the model of NB, μ_i always equals to 1 since NB will not be biased to the latest data subset. For MLPs, μ_i is updated once the MLP is selected to be trained. For the MLPs which were not selected to be trained and the model of NB, N_{km} from the new data subset can be accumulated to the previous one to update R_k and P_k and then update w_{ik}. In this way, w_{ik} is updated not according to the current data subsets only and it would be helpful for preserving previous information.

The pseudo-code for the approach is presented in Fig. 1. In the pseudo-code, Select is the selecting process for selecting MLPs from the ensemble to be trained with the new data sub-set, MLPs-Training and NB-Training were the training process for training the MLPs and NB in the ensemble. The details of these processes are described in the following subsection.

3.2. **Some details inside SFL.**

3.2.1. *Selecting Process.* The selecting process is based on the current data subset S_t. The individuals are added to Ens_{sel} one by one by greedy strategy. Every time, every MLP in Ens_{res} is temporarily added to Ens_{sel} to estimate the performance (i.e., the arithmetical mean F-measures of all classes) on the current data subset. The MLPs in Ens_{sel} are tested one by one and the MLP that makes Ens_{sel} perform the worst will be finally added to Ens_{sel}

If the current data subset does not contain some classes that have appeared in the previous data subsets, the selection process should ensure that not all the MLPs that have been trained with the data of the lost classes are added to Ens_{sel}.

Therefore, when an MLP is added to Ens_{sel}, the following constraint should be satisfied for the MLPs in Ens_{sel}:

$$\Pi_{k\in L}(\sum_i w_{ik}) \neq 0. \tag{7}$$

where $L = \{k|\text{“class } k \text{ is not contained in } S_t\text{”}\}$. If there is no MLP that can be added to Ens_{sel}, a new initialized MLP will be generated and added to Ens_{sel}.

In this way, the MLPs that are not well trained are selected to be further trained. Besides, the MLPs that are reserved in Ens_{res} could preserve the previously learned information.

3.2.2. *Training the model of Naive Bayes.* According to Bayes Decision Theory, the probability of an testing example $x = \{x_i|i = 1, 2, ..., d\}$ belonging to class k is

$$\mathcal{P}(k|x) = \frac{\mathcal{P}(x|k)\mathcal{P}(k)}{\mathcal{P}(x)} = \frac{\mathcal{P}(x|k)\mathcal{P}(k)}{\sum_{k=1}^{C}\mathcal{P}(x|k)\mathcal{P}(k)} \tag{8}$$

where $\mathcal{P}(kx)$ is the posterior probability of an examples x belonging to class k, C is the number of classes and $\mathcal{P}(k)$ is the prior probabilities of class k. In class imbalance situation, we assume that $\mathcal{P}(k)$ are equal for all of the classes. Besides, all the features of the examples are assumed to be independent to each other. Therefore, the probability of (8) becomes

$$\mathcal{P}(k|x) = \frac{\Pi_{i=1}^{d}\mathcal{P}(x_i|k)}{\sum_{k=1}^{C}\Pi_{i=1}^{d}\mathcal{P}(x_i|k)}. \tag{9}$$

In incremental learning mode, $\mathcal{P}(x_i|k)$ is updated as every new data subset comes up.$\mathcal{P}(x_i|k)$ can be estimated in the form of $n(x_ik)/n(k)$, where $n(x_ik)$ is the number of examples that belongs to class k and the value of its ith feature is x_i, $n(k)$ is the number of examples that belongs to class k. Both $n(x_ik)$ and $n(k)$ can be estimated in each data subset and then accumulated to estimate $\mathcal{P}(x_i|k)$. In this way, NB can learn from new data subsets without any loss of previous information.

The estimation of $\mathcal{P}(kx)$ in (9) requires the values of features to be discrete. Specifically, for the features with continuous values, average partition is used to discretize the features for calculating $\mathcal{P}(x_i|k)$.

3.2.3. *Training the ensemble of MLPs.* We have proposed a Dynamic Sampling (DyS) method for class imbalance problems [13], which can be used for training the ensemble of MLPs. Similarly to the approach proposed in [13], the main process of DyS for an ensemble is presented as follows (in one epoch):
step1. Randomly fetch an example x from the training set;
step2. Estimate the probability p that the example should be used for updating the ensemble.
step3. Generate a uniform random real number μ between 0 and 1.
step4. If $\mu < p$, then use x to update the ensemble using Negative Correlation Learning (NCL) [23] to make every MLP negatively correlated to other individuals (including the MLPs in Ens_{res} and the model of NB).
step5. Repeat steps 1 to 3 until there is no example in the training set.

The above steps will be repeated until stop criterion is satisfied. The following shows the method for estimating p, which was the main issue in DyS.

In a problem with nc classes, we set nc output nodes for all of the MLPs and for an example belonging to class k, we set the target output of the example as $t = \{t_i|t_k = 1, t_{(j|\neq k)} = 0\}$. The real output of the example is denoted as $y =$

$\{y_i|i = 1, 2, ..., nc\}$, so the node with the highest output designates the class. Both the hidden node functions and the output node functions of all MLPs are set as the logistic function $\varphi(x) = 1/(1 + e^{-x}), so \quad that \quad y_i \in (0, 1)$.

The same to [13], the probability that an example belonging to class k will be used to update the ensemble is estimated as:

$$\text{p}=\begin{cases} 1, & \text{if } \delta < 0, \\ e^{\frac{-\delta r_k}{min_i\{r_i\}}}, & \text{otherwise} \end{cases} \tag{10}$$

where $\delta = y_k - max_{i \neq k}\{y_i\}$ is the confidence of the current ensemble correctly classifying the example. For more details of DyS, please refer to [13].

By employing DyS, the MLPs in the ensemble are able to accommodate to class imbalance situations.

3.3. **The reason for the success of SFL.** In general, there are several essentials inside SFL that would make SFL successful, including the selective training of MLPs, the use of NB, the setting of impact weights for comb-ing the individuals in the ensemble, and the consideration of class imbalance in training process.

To analyze the reason for the success of SFL, two especial cases in incremental learning are considered, i.e., new classes in the new data subsets and the loss of previous classes in the new data subsets. The ensemble are divided into three parts: MLP_a, MLP_b and NB, where MLP_a is the MLPs that are selected for learning the new data subset and MLP_b is the rest MLPs.

After learning a new data subset which contains new classes, MLP_a and NB have learnt the new classes while MLP_b have not. In SFL, the impact weights for MLP_b is updated by accumulating the performance of MLP_b on previous data subsets and the current data subset. The examples of the new classes will be misclassified as other classes by MLP_b. This will cause the degradation of precision of MLP_b on those classes and finally degrade the impact weights of MLP_b on those classes. In this way, the wrong prediction of MLP_b on a testing example of new classes would impact less on the prediction of the whole ensemble. Therefore, MLP_a and NB which have learnt the new classes will play a leading role in the ensemble when predicting the examples of new classes.

After learning a new data subset which loses some previous classes, MLP_a will be biased to the classes that are contained in the current data subset. However, the impact weights of MLP_a will be degraded by a coefficient according to (5). Therefore, NB and the MLPs which is recently trained with the new classes will play a leading role in the ensemble when making the prediction.

Besides, as we discuss before, NB is able to learn incrementally without forgetting previous information. The use of NB will help to prevent the ensemble from catastrophic forgetting. Furthermore, in the training of NB and MLPs, the situation of class imbalance is considered. Therefore, SFL is able to deal with class imbalance in the new data subsets.

4. **Experimental study.** To assess the performance of SFL, some synthetic data sets and real-world data sets were used to conduct the experiments. First of all, three types of synthetic data sets were generated to simulate the incremental learning process. Then, 5 real-world data sets with imbalanced class distributions from UCI repository [1] were used to simulate incremental learn-ing by randomly dividing the data sets. Finally, another 5 real-world data sets from UCI repository, including 3 class imbalanced data sets and 2 class balanced data sets, were used to simulate

the incremental learning process by dividing the data sets. In this part, the dividing of the data sets considered new classes and the loss of previous classes in the new data subsets. The purpose of this part of experiment is to assess the ability of SFL learning form new classes and preserving previous in-formation when some classes are lost in the new data subsets. As a recently proposed approach which also addressed for class imbalanced incremental learning, Learn++.UDNC [5] was used for the comparison. Besides, in order to find out the efficiency of MLPs and NB to SFL, the model of ensembles with only MLPs (referred as SFL.MLP) and the model of NB are also compared with SFL. The recall of every class and the arithmetic mean values over recalls of all classes are used as the metric.

TABLE I THE DATA DISTRIBUTIONS OF THE SYNTHETIC DATA SETS, WHERE C_i DENOTES CLASS i, S_i DENOTES THE iTH DATA SUBSET

	Type A				Type B				Type C			
	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4
S_1	500	500	500	0	500	10	0	0	500	10	0	0
S_2	500	500	500	10	500	10	10	0	0	500	10	0
S_3	10	500	500	10	500	10	10	10	0	0	500	10
S_4	500	10	500	10	500	0	10	10	10	0	0	500
S_5	500	500	10	10	500	0	0	10	–	–	–	–
Test	100	100	100	100	100	100	100	100	100	100	100	100

FIGURE 2. This is Table 1

4.1. **Experiments on synthetic data sets.** The synthetic data were generated as follows. Data of four 2-dimensional Normal Distributions were generated for four classes. The means were $\mu_1 = (0,0), \mu_2 = (0,1), \mu_3 = (1,1) and \mu_4 = (1,0)$, the two features are independent with variances $\sigma_1 = \sigma_2 = \sigma_3 = \sigma_4 = 0.2$. Three types of synthetic data sets were generated. TABLE I presents the class distributions of every data subset for the three types. In Type A, there are three majority classes (class 1 to 3) and one minority class (class 4). Class 4 comes up as a new class in S_2. Class 1 to 3 appears to be another minority class in training subsets S_3 to S_5, respectively. This experiment was conducted to see the performance of SFL on problems with multi majority classes (which appear to be minority classes sometimes) and single minority class (also comes up as new class). In Type B, there are one majority class and three minority classes. Class 2 comes up at the beginning but is lost in the last two training subsets. Class 3 comes up as a new class in S_2 and is lost in the last training subset. Class 4 comes up as a new class in S_3. This experiment was conducted to see the performance of SFL on problems with single majority class and multi minority classes, some of which come up as new classes and are lost in some data subsets. In Type C, the class distribution of the whole training set (i.e., the union of all the training subsets) is balanced. However, the training subsets are class imbalanced and every training subset contains only two classes. This experiment was conducted to see the performance of SFL on problems whose class distributions are balanced in total but imbalanced in data subsets. The distributions are quite different between the data subsets in all the three types.

10 MLPs with 20 hidden nodes of every MLP was used in SFL and SFL.MLP. The training stop error was 0.05 and the coefficient of the penalty term of NCL (referred as λ) was 0.5. The data sets were generated 30 times independently, and the means and standard deviations over 30 executions of the three types of data sets are presented in TABLE II, TABLE III and TABLE IV, respectively.

TABLE II THE RECALLS (%) OF EVERY CLASS ON TESTING SET OVER 30 EXECUTIONS ON TYPE A OF SYNTHETIC DATA SET, WHERE AVG DENOTES THE ARITHMETICAL AVERAGE OF THE RECALLS OF ALL THE CLASSES. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

		C_1	C_2	C_3	C_4	AVG
SFL	S_1	87.70±4.85	68.30±8.80	85.30±5.39	0	60.33±1.98
	S_2	81.30±4.33	71.83±6.72	79.73±5.04	56.87±8.78	72.43±2.80
	S_3	72.90±6.57	76.60±6.93	79.63±5.94	62.30±9.83	72.86±2.62
	S_4	78.40±6.16	71.53±6.94	81.63±5.04	62.57±8.34	73.53±2.72
	S_5	81.73±6.71	75.10±7.02	73.80±7.55	61.83±7.49	73.12±2.75
Learn++.UDNC	S_1	83.47±6.80	**73.93±8.59**	82.00±6.97	0	59.85±2.23
	S_2	**83.90±4.71**	**75.57±5.70**	**82.67±4.99**	15.03±10.82	64.29±3.28
	S_3	**81.40±5.14**	77.70±5.31	82.20±4.92	20.53±10.25	65.46±2.97
	S_4	**82.50±5.49**	**75.67±5.50**	83.30±4.86	26.17±9.47	66.91±3.04
	S_5	82.30±5.52	**77.90±5.24**	**80.37±6.04**	28.63±8.59	67.30±2.82
SFL.MLP	S_1	88.40±6.08	58.40±13.63	**88.90±7.48**	0	58.93±2.50
	S_2	**85.10±5.52**	66.23±9.31	**84.70±6.69**	48.73±6.36	71.19±2.75
	S_3	71.57±6.70	77.60±8.10	79.40±8.60	60.73±8.47	72.32±2.86
	S_4	78.20±6.18	72.13±8.11	81.70±6.22	61.07±8.49	73.27±2.42
	S_5	79.97±5.49	**77.30±6.95**	73.60±7.34	62.40±8.16	73.32±2.55
NB	S_1	83.90±4.80	**73.63±5.73**	84.17±6.53	0	60.43±2.14
	S_2	74.47±8.43	72.73±6.02	77.03±8.58	57.27±15.44	70.38±4.07
	S_3	73.73±7.88	72.67±6.05	78.03±7.47	**66.33±9.05**	72.69±3.32
	S_4	73.70±6.89	72.70±6.08	77.53±7.85	**69.47±9.54**	73.35±3.05
	S_5	73.93±7.34	72.47±6.28	**77.13±8.44**	**70.93±8.77**	73.62±3.08

FIGURE 3. This is table2

TABLE III THE RECALLS (%) OF EVERY CLASS ON TESTING SET OVER 30 EXECUTIONS ON TYPE B OF SYNTHETIC DATA SET, WHERE AVG DENOTES THE ARITHMETICAL AVERAGE OF THE RECALLS OF ALL THE CLASSES. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

		C_1	C_2	C_3	C_4	AVG
SFL	S_1	90.67±4.85	74.63±11.44	0	0	41.33±2.38
	S_2	88.33±3.86	67.20±10.53	76.00±11.49	0	57.88±2.85
	S_3	81.10±6.26	64.90±6.79	74.30±7.82	63.30±13.47	70.90±2.46
	S_4	82.20±6.11	56.43±9.56	77.23±8.54	68.37±10.23	71.06±2.38
	S_5	84.27±3.77	56.53±9.15	71.07±9.31	72.27±9.89	71.03±2.35
Learn++.UDNC	S_1	73.00±16.79	**90.87±4.31**	0	0	40.97±3.92
	S_2	85.40±6.54	69.33±10.14	66.97±11.60	0	55.42±2.87
	S_3	**86.97±6.71**	62.33±10.94	63.30±10.81	37.00±18.49	62.40±6.78
	S_4	**89.00±4.45**	54.87±10.90	69.17±6.22	45.97±12.29	64.75±4.65
	S_5	**91.97±4.06**	47.60±11.06	67.33±7.21	52.40±9.56	64.83±3.90
SFL.MLP	S_1	88.60±6.17	**79.50±15.64**	0	0	**42.02±2.98**
	S_2	87.70±5.05	69.07±13.82	62.30±24.49	0	54.77±4.31
	S_3	**85.30±4.73**	65.00±8.33	73.93±7.80	60.60±10.55	71.21±2.24
	S_4	**85.50±4.52**	53.13±12.65	76.63±8.31	67.83±8.63	70.78±2.24
	S_5	**86.90±4.35**	42.67±19.17	70.73±10.85	73.47±8.52	68.44±3.94
NB	S_1	86.27±6.71	63.87±13.68	0	0	37.53±3.49
	S_2	83.33±5.47	64.77±7.90	62.30±13.00	0	52.60±3.22
	S_3	74.17±6.61	67.03±7.13	68.47±8.23	54.30±14.63	65.99±4.56
	S_4	73.43±8.17	**67.53±7.64**	69.63±7.72	66.07±8.81	69.17±2.47
	S_5	71.80±7.19	**67.73±7.69**	69.67±6.89	69.73±9.07	69.73±2.04

FIGURE 4. This is table3

Wilcoxon signed-rank test with the level of significance $\alpha = 0.05$ was employed for the comparison between SFL and other methods. In the results of other methods, the values with underline (or bold) denote that SFL performed significantly better (or worse) than them on those values and the values with normal type denote that there are no significant differences. The results on Type A data set are presented in TABLE II. Comparing to Learn++.UDNC, SFL gets better overall recalls (i.e., the average of the recalls of all classes). The recalls of SFL on class 1 to class 3, which are

TABLE IV THE RECALLS (%) OF EVERY CLASS ON TESTING SET OVER 30 EXECUTIONS ON TYPE C OF SYNTHETIC DATA SET, WHERE AVG DENOTES THE ARITHMETICAL AVERAGE OF THE RECALLS OF ALL THE CLASSES. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

		C_1	C_2	C_3	C_4	AVG
SFL	S_1	93.17±4.24	72.03±12.52	0	0	41.30±2.68
	S_2	82.47±6.78	79.57±5.78	69.60±12.18	0	57.91±2.87
	S_3	68.90±12.34	72.80±6.78	81.20±5.75	57.53±12.44	70.11±2.81
	S_4	70.93±7.97	70.30±8.02	73.43±8.23	79.63±6.72	73.57±2.30
Learn++.UDNC	S_1	74.67±14.70	**87.40±7.47**	0	0	40.52±3.22
	S_2	74.67±14.70	65.43±17.19	71.73±15.62	0	52.96±4.33
	S_3	74.67±14.70	65.33±17.26	50.27±20.92	18.10±25.05	52.09±7.38
	S_4	66.60±12.07	67.20±16.77	58.73±20.78	63.07±18.60	63.90±6.94
SFL.MLP	S_1	88.37±6.05	**82.43±9.19**	0	0	**42.70±1.76**
	S_2	56.07±14.04	**83.50±7.04**	**77.03±8.50**	0	54.15±3.42
	S_3	0	73.93±7.87	**85.83±4.89**	**74.07±7.74**	58.46±2.44
	S_4	72.83±7.87	0.60±2.93	69.17±8.30	**87.17±5.51**	57.44±1.99
NB	S_1	90.53±5.86	61.70±13.31	0	0	38.06±2.77
	S_2	**84.23±6.33**	76.30±7.67	61.97±11.75	0	55.63±2.95
	S_3	**75.50±9.07**	73.07±8.80	75.83±8.34	56.50±11.61	70.22±2.79
	S_4	**74.07±7.92**	**73.27±8.75**	74.50±6.85	74.50±8.19	74.08±2.63

FIGURE 5. This is table4

TABLE V THE CLASS DISTRIBUTIONS OF 5 RANDOMLY DIVIDED DATA SETS

Data sets	#C	Class distributions
Balance-scale	3	49: 288: 288
Soybean	17	20: 20: 20: 88: 44: 20: 20: 92: 20: 20: 20: 44: 20: 91: 91: 15: 16
Splice	3	767: 768: 1655
Thyroid-allrep	4	3648: 38: 52: 34
Thyroid-ann	3	166: 368: 6666

FIGURE 6. This is table5

majority classes, are not as good as Learn++.UDNC. However, Learn++.UDNC is biased too much to the majority classes and performs very poor on the only minority class while SFL performs more balanced over all the classes. Therefore, SFL outperforms Learn++.UDNC in this data set. Comparing to SFL.MLP and NB, there is few statistical difference on average recalls, especially after training with S_3, S_4 and S_5. After training with S_2, where class 4 comes up as a new class, SFL learns better of class 4 than SFL.MLP and as good as NB. At the same time, SFL does not degrade as much performance on class 1 as NB does. Although SFL degrades more performance on class 1 and class 3 than SFL.MLP, it performs better than SFL.MLP on class 2. Therefore, after training with S_2, SFL performs better than both SFL.MLP and NB on the average recall. This observation indicates that SFL is capable of combining the advantages of both MLPs and NB to make a better model.

The results on Type B data set are presented in TABLE III. When comparing to Learn++.UDNC, the similar observations can be made and we can also conclude that SFL outperforms Learn++.UDNC in this data set. When comparing to SFL.MLP and NB, some values of SFL are between the values of SFL.MLP and NB (always closer to the larger ones), some values of SFL are significantly larger than both SFL.MLP and NB. Observing the results on Type C data set in TABLE IV,

TABLE VI THE DATA DISTRIBUTIONS OF THE DATA SUBSETS AND TESTING SET, WHERE C_i DENOTES CLASS i, S_i DENOTES THE iTH DATA SUBSET

(a) Class imbalanced data sets

	Car				*Nursery*				*Page-blocks*				
	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4	C_1	C_2	C_3	C_4	C_5
S_1	200	100	0	20	1500	100	0	500	2000	80	0	20	30
S_2	500	80	20	20	100	1000	50	800	1000	80	10	20	20
S_3	200	100	20	0	500	500	50	1000	1000	80	10	20	30
S_4	–	–	–	–	500	500	0	100	–	–	–	–	–
S_5	–	–	–	–	100	500	50	100	–	–	–	–	–
S_6	–	–	–	–	100	200	50	100	–	–	–	–	–
Test	310	104	25	29	1466	1520	128	1444	913	89	8	28	35

(b) Class balanced data sets

	Optdigits										*Vehicle*			
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_1	C_2	C_3	C_4
S_1	380	0	20	0	0	380	20	0	0	380	150	0	0	10
S_2	0	0	380	20	0	0	380	380	0	0	0	150	10	0
S_3	0	20	0	380	0	0	0	20	380	0	10	0	150	0
S_4	20	380	0	0	400	20	0	0	20	20	0	10	0	140
Test	154	171	157	172	168	158	158	166	154	162	52	57	58	49

FIGURE 7. This is table6

the similar observations can be made. All these results show that SFL outperforms Learn++.UDNC and is capable of combining the advantages of both MLPs and NB to make a better model.

4.2. **Experiments on real-world data sets.** The experiments on real-world data sets include three parts. First of all, 5 class imbalanced data sets were divided randomly to simulate the incremental learning process. Secondly, 3 class imbalanced data sets were divided with considering new classes and the loss of classes in the new data subsets. Finally, 2 class balanced data sets were divided into some class imbalanced subsets to simulate the incremental learning process. The situations of new classes and the loss of classes in the new data subsets were also considered.

The class distributions of the 5 class imbalanced data sets that were randomly divided are presented in TABLE V. Each one of these data sets was firstly stratified divided into training set (80%) and testing set (20%) and then the training set was randomly divided into 5 training subsets. The other real-world data sets, including 3 class imbalanced data sets and 2 class balanced data sets were divided according to predefined data distributions. The data distributions of all training subsets and testing sets were presented in TABLE VI. It can be observed from TABLE VI that for all the data sets, the data distributions between different training subsets are quite different and the situations of new classes and the loss of classes in the new data subsets occur in some training subsets.

For all the data sets, 10 MLPs with 20 hidden nodes of every MLP was used in SFL and the coefficient λ of NCL was 0.5. An independent execution was implemented for every data set to set the stop criterion for training MLPs to ensure the convergence of the training process. All the data sets were divided 30 times independently and for every time, all the comparing methods were executed once. The means and standard deviations of the overall recalls after every coming up of data subset over 30 executions of all the real-world data sets are presented in TABLE VII. Wilcoxon signed-rank test with the level of significance $\alpha = 0.05$ was employed

TABLE VII THE OVERALL RECALLS (%)ON TESTING SET OVER 30 EXECUTIONS ON REAL-WORLD DATA SETS. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

Data set	Method	S_1	S_2	S_3	S_4	S_5	S_6
Balance-scale	SFLHE	80.12±6.99	84.76±6.34	86.41±5.28	88.26±4.81	87.51±6.57	–
	Learn++.UDNC	77.15±6.73	84.70±6.60	86.82±5.35	87.89±5.05	88.11±5.07	–
	SFL.MLP	80.09±6.98	84.51±6.60	86.28±6.00	88.73±4.17	88.58±5.58	–
	NB	57.17±6.52	60.75±5.54	63.79±4.76	64.37±5.90	67.18±7.33	–
Soybean	SFLHE	85.70±6.07	91.60±2.62	93.69±1.98	94.55±1.45	94.70±1.73	–
	Learn++.UDNC	74.01±7.80	80.92±4.95	85.10±3.48	87.13±2.63	88.92±2.66	–
	SFL.MLP	85.76±5.59	90.77±3.24	92.38±2.14	92.92±1.65	93.14±1.82	–
	NB	38.87±5.02	63.87±6.18	79.03±5.14	87.43±3.46	92.27±2.65	–
Splice	SFLHE	79.84±2.10	83.37±1.61	83.63±1.88	84.71±1.61	84.56±1.62	–
	Learn++.UDNC	71.40±2.65	75.89±2.47	78.19±2.20	79.42±1.76	79.98±1.73	–
	SFL.MLP	77.69±1.89	80.50±1.37	81.00±1.62	81.47±1.15	81.98±1.83	–
	NB	**94.35±1.06**	**94.95±0.84**	**95.08±0.77**	**95.24±0.74**	**95.27±0.63**	–
Thyroid-allrep	SFLHE	61.46±10.15	66.79±7.92	72.35±6.64	74.58±7.00	75.94±6.57	–
	Learn++.UDNC	63.34±7.55	63.00±8.90	59.25±8.89	57.25±7.16	60.43±7.34	–
	SFL.MLP	64.33±10.40	**70.21±7.47**	72.33±7.69	74.10±5.58	74.54±7.08	–
	NB	38.97±7.21	52.86±9.13	63.33±7.48	68.17±6.93	73.19±6.66	–
Thyroid-ann	SFLHE	79.03±5.79	81.67±5.16	83.39±4.69	85.06±4.41	85.75±4.57	–
	Learn++.UDNC	**84.44±3.11**	**86.53±2.43**	**85.90±3.01**	86.15±2.82	85.83±2.82	–
	SFL.MLP	78.67±4.95	81.85±4.73	84.07±5.17	84.72±3.75	87.21±3.57	–
	NB	74.20±5.01	79.06±2.94	80.21±2.38	80.57±2.11	80.83±2.08	–
Car	SFLHE	63.25±2.95	84.78±2.56	85.26±2.86	–	–	–
	Learn++.UDNC	59.37±3.94	76.98±7.29	79.83±5.50	–	–	–
	SFL.MLP	57.97±3.05	77.19±3.75	77.72±3.69	–	–	–
	NB	**64.91±2.34**	84.92±2.18	86.32±1.93	–	–	–
Nursery	SFLHE	74.09±0.28	94.72±1.74	97.29±0.80	94.09±2.25	97.43±0.62	97.39±0.58
	Learn++.UDNC	72.66±0.86	73.06±1.03	80.14±8.02	77.68±6.70	83.35±7.88	88.45±5.40
	SFL.MLP	74.05±0.28	89.39±6.15	96.51±0.90	84.31±11.17	96.01±1.86	96.20±1.25
	NB	68.49±0.58	90.18±1.06	90.85±0.61	90.82±0.57	90.93±0.56	90.99±0.57
Page-blocks	SFLHE	68.14±1.99	84.90±5.49	87.36±3.42	–	–	–
	Learn++.UDNC	68.53±3.50	86.70±3.78	88.35±1.89	–	–	–
	SFL.MLP	68.23±2.67	84.29±4.66	86.30±3.21	–	–	–
	NB	54.81±2.24	57.53±2.89	61.73±2.71	–	–	–
Optdigits	SFLHE	49.32±0.21	64.62±0.85	80.13±0.66	92.91±0.58	–	–
	Learn++.UDNC	48.13±1.40	55.80±3.59	59.38±3.63	76.47±4.83	–	–
	SFL.MLP	49.36±0.20	54.77±1.16	59.95±8.84	80.45±7.22	–	–
	NB	34.54±1.49	59.35±0.81	76.67±0.76	91.01±0.69	–	–
Vehicle	SFLHE	44.69±2.07	44.43±4.97	53.98±3.99	66.58±2.31	–	–
	Learn++.UDNC	44.21±2.61	40.65±3.65	53.74±5.64	60.37±5.64	–	–
	SFL.MLP	44.95±2.18	**47.09±6.05**	46.54±4.21	57.56±6.66	–	–
	NB	28.59±2.00	34.42±3.19	47.45±2.76	61.70±2.18	–	–

FIGURE 8. This is table7

for the comparison between SFL and other methods. In the results of other methods, the values with underline (or bold) denote that SFL performed significantly better (or worse) than them on those values and the values with normal type denote that there are no significant differences.

It can be observed form TABLE VII that SFL can outperform Learn++.UDNC on most of the data sets, including Soybean, Splice, Thyroid-allrep, Car, Nursery, Optdigits and Vehicle. On the other data sets, SFL also does not perform significantly worse than Learn++.UDNC. When comparing with SFL.MLP and NB, the performance of SFL usually leans to the better one of SFL.MLP and NB and sometimes SFL outperforms both of them, such as the performance on Soybean, Nursery, Optdigits and Vehicle. These observations go a step further to support

TABLE VIII THE RECALLS (%) OF EVERY CLASS ON TESTING SET OVER 30 EXECUTIONS ON *NURSERY*, WHERE AVG DENOTES THE ARITHMETICAL AVERAGE OF THE RECALLS OF ALL THE CLASSES. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

		C_1	C_2	C_3	C_4	AVG
SFL	S_1	98.90±0.46	100.00±0.00	0	97.44±0.92	74.09±0.28
	S_2	93.75±1.44	100.00±0.00	87.27±7.31	97.87±1.02	94.72±1.74
	S_3	96.76±1.30	100.00±0.00	94.74±3.71	97.67±1.09	97.29±0.80
	S_4	98.62±0.86	100.00±0.00	83.20±8.91	94.53±2.06	94.09±2.25
	S_5	94.77±2.41	100.00±0.00	98.18±2.64	96.76±1.36	97.43±0.62
	S_6	94.36±2.09	100.00±0.00	98.70±2.32	96.51±1.60	97.39±0.58
Learn++.UDNC	S_1	95.24±1.32	98.58±3.38	0	96.81±0.99	72.66±0.86
	S_2	**94.43±1.50**	99.89±0.17	1.28±4.50	96.64±0.92	73.06±1.03
	S_3	95.02±1.43	99.99±0.04	28.05±32.63	97.53±0.63	80.14±8.02
	S_4	96.04±1.12	99.99±0.03	18.70±27.44	**96.01±0.98**	77.68±6.70
	S_5	95.53±1.33	100.00±0.01	42.03±32.06	95.85±0.96	83.35±7.88
	S_6	95.10±1.47	100.00±0.01	62.81±22.16	95.89±0.96	88.45±5.40
SFL.MLP	S_1	98.99±0.55	100.00±0.00	0	97.21±1.11	74.05±0.28
	S_2	74.82±15.10	100.00±0.00	85.76±29.18	96.98±1.55	89.39±6.15
	S_3	90.87±4.75	100.00±0.00	**97.66±2.94**	97.52±1.12	96.51±0.90
	S_4	94.33±7.03	100.00±0.00	48.54±49.44	94.38±2.31	84.31±11.17
	S_5	89.76±5.82	100.00±0.00	97.03±7.93	97.27±1.09	96.01±1.86
	S_6	89.11±5.46	100.00±0.00	98.41±3.01	**97.30±1.35**	96.20±1.25
NB	S_1	86.98±1.40	100.00±0.00	0	86.96±1.63	68.49±0.58
	S_2	77.66±1.80	100.00±0.00	**95.44±4.35**	87.62±1.18	90.18±1.06
	S_3	77.58±1.25	100.00±0.00	**97.73±2.26**	88.08±1.06	90.85±0.61
	S_4	77.64±1.27	100.00±0.00	**97.71±2.12**	87.94±1.04	90.82±0.57
	S_5	77.42±1.19	100.00±0.00	98.33±1.82	87.97±1.00	90.93±0.56
	S_6	77.64±1.36	100.00±0.00	98.33±1.89	88.00±1.01	90.99±0.57

FIGURE 9. This is table8

TABLE IX THE RECALLS (%) OF EVERY CLASS ON TESTING SET OVER 30 EXECUTIONS ON *OPTDIGITS*, WHERE AVG DENOTES THE ARITHMETICAL AVERAGE OF THE RECALLS OF ALL THE CLASSES. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR THE COMPARISON BETWEEN SFL AND OTHER METHODS. IN THE RESULTS OF OTHER METHODS, THE VALUES WITH UNDERLINE (OR BOLD) DENOTE THAT SFL PERFORMED SIGNIFICANTLY BETTER (OR WORSE) THAN THEM ON THOSE VALUES AND THE VALUES WITH NORMAL TYPE DENOTE THAT THERE ARE NO SIGNIFICANT DIFFERENCES.

		C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	AVG
SFL	S_1	99.5±0.6	0	98.1±1.8	0	0	98.0±0.8	99.2±0.8	0	0	98.4±0.7	49.3±0.2
	S_2	99.4±0.5	0	98.5±1.3	74.7±7.9	0	88.8±4.2	99.3±0.7	98.3±0.8	0	87.2±5.5	64.6±0.8
	S_3	98.1±1.1	50.1±5.4	94.1±1.6	92.4±2.3	0	88.2±3.3	98.4±1.0	98.4±0.8	93.2±2.2	88.3±2.3	80.1±0.7
	S_4	98.8±0.7	91.6±1.8	90.5±2.2	89.0±2.8	93.0±2.0	95.3±2.1	96.7±1.4	96.7±1.5	89.8±2.5	87.6±2.8	92.9±0.6
Learn++. UDNC	S_1	98.7±1.4	0	96.8±2.2	0	0	96.9±1.8	99.0±0.7	0	0	96.2±1.3	48.8±0.3
	S_2	94.8±6.3	0	96.5±2.1	49.9±12.4	0	86.8±6.8	98.3±3.2	79.6±24.9	0	69.8±8.4	57.6±2.4
	S_3	86.1±20.3	**54.2±6.6**	82.5±6.5	49.8±12.3	0	88.3±5.3	26.6±19.2	48.9±15.7	82.1±6.2	76.6±6.7	59.5±3.9
	S_4	91.2±14.2	83.2±5.5	92.0±4.3	89.1±6.5	91.6±4.8	65.7±12.5	53.8±20.7	79.2±8.2	75.9±6.8	49.9±11.3	77.2±2.7
SFL.MLP	S_1	99.5±0.7	0	98.3±1.8	0	0	98.1±0.9	99.3±0.8	0	0	98.4±1.0	49.4±0.2
	S_2	97.0±1.8	0	98.5±1.6	**97.2±1.2**	0	39.9±9.1	99.4±0.7	92.5±5.7	0	23.3±6.0	54.8±1.2
	S_3	26.5±38.6	**88.3±6.2**	94.6±2.7	**97.1±1.6**	0	22.2±30.6	70.3±16.9	98.0±1.7	86.9±6.7	15.6±21.9	59.9±8.8
	S_4	98.9±0.9	90.0±8.4	58.0±42.9	52.6±9.9	79.3±16.1	**96.4±1.8**	63.9±46.1	89.5±5.3	88.1±4.5	87.8±3.8	80.5±7.2
NB	S_1	98.6±1.2	0	24.5±8.1	0	0	89.5±2.9	36.9±11.9	0	0	95.8±1.7	34.5±1.5
	S_2	98.6±1.2	0	95.0±1.6	23.5±6.8	0	89.2±3.0	97.5±1.2	96.8±1.3	0	**92.8±2.4**	59.4±0.8
	S_3	98.2±1.2	24.0±6.3	90.9±1.8	88.6±2.3	0	88.3±3.0	97.4±1.2	96.8±1.5	93.4±2.0	**88.9±2.4**	76.7±0.8
	S_4	97.8±1.2	86.5±2.4	90.3±2.0	88.6±2.4	91.0±2.3	87.6±3.0	96.4±1.3	95.8±1.5	89.2±2.1	87.0±2.5	91.0±0.7

FIGURE 10. This is table9

that SFL is capable of combing the advantages of both MLPs and NB to make a better model.

On Car, Nursery, Page-blocks, Optdigits and Vehicle, the data sets were divided according the distribution presented in TABLE VI, where coming up new classes or losing previous classes usually occurs in the new data subsets. It will be worthy to

TABLE X THE COMPUTATIONAL TIME (IN SECONDS) OF SFL AND LEARN++.UDNC ON ALL THE DATA SETS

	SFL	Learn++.UDNC
Synthetic Type A	217	2116
Synthetic Type B	31	680
Synthetic Type C	17	489
Car	58	69
Nursery	303	508
Page-blocks	129	749
Optdigits	84	66
Vehicle	52	78
Balance-scale	93	145
Soybean	76	117
Splice	252	542
Thyroid-allrep	78	773
Thyroid-ann	186	1060

FIGURE 11. This is table10

see the detailed results of each class on these data sets. Therefore, the detailed results on two of them, i.e., Nursery (class imbalanced) and Optdigits (class balanced) were further presented.

The means and standard deviations over 30 executions of Nursery are presented in TABLE VIII. Wilcoxon signed-rank test with the level of significance $\alpha = 0.05$ was employed for the comparison between SFL and other methods. In the results of other methods, the values with underline (or bold) denote that SFL performed significantly better (or worse) than them on those values and the values with normal type denote that there are no significant differences. It can be observed from TABLE VIII that, the performance of Learn++.UDNC on class 3 is much worse than that of SFL. Class 3 is a minority class. It comes up in S_2 as a new class and is lost in S_4. The observations indicate that SFL could handle this kind of problems. To see the effect of MLPs and NB in SFL, we pay more attentions to the comparison to SFL.MLP and NB. It can be observed that, MLPs could learn better than NB if there is not any class loss. However, when class 3 is lost in S_4, MLPs degrades much more recall on class 3 than NB. As the combination of MLPs and NB, SFL does not degrade too much recall on class 3 and at the same time, SFL learns better than NB on other classes, which leads to the better overall recalls. Therefore, in this data set, MLPs help SFL to learn better and NB helps SFL to preserve the previously learned information especially when class loss occurs.

The means and standard deviations over 30 executions of Optdigits are presented in TABLE IX. Wilcoxon signed-rank test with the level of significance $\alpha = 0.05$ was employed for the comparison between SFL and other methods. In the results of other methods, the values with underline (or bold) denote that SFL performed significantly better (or worse) than them on those values and the values with normal type denote that there are no significant differences. In S_2, class 4 first comes up as a minority class, class 8 first comes up as a majority class, class 1, class 6 and class 10 are lost. After learning from S_2, the recall of class 4 of SFL is larger than those of Learn++.UDNC and NB, but not as large as SFL.MLP; the recall of class 8 of SFL is the best of all others. At the same time, the degradation of class 1, class 6 and class 10 of SFL is much less than Learn++.UDNC and SFL.MLP and a bit more than NB. This observation indicates SFL can learn new classes with little performance degradation of other lost classes. Even though SFL.MLP can perform

TABLE XI THE ESTIMATIONS OF THE RATIOS (%) ACCORDING TO (11).

		S_1	S_2	S_3	S_4	S_5	S_6
Synthetic Type A	ρ_1	8.13	17.37	15.71	13.89	13.92	–
	ρ_2	45.64	55.82	45.11	47.15	47.33	–
	ρ_3	54.36	44.18	54.89	52.85	52.67	–
	ρ_4	53.74	56.13	53.20	54.63	51.66	–
Synthetic Type B	ρ_1	11.62	16.31	20.08	19.32	23.80	–
	ρ_2	67.22	65.58	63.19	51.96	44.87	–
	ρ_3	32.78	34.42	36.81	48.04	55.13	–
	ρ_4	64.35	64.44	59.51	57.39	59.32	–
Synthetic Type C	ρ_1	10.27	16.44	34.47	33.87	–	–
	ρ_2	69.62	55.03	32.16	23.36	–	–
	ρ_3	30.38	44.97	67.84	76.64	–	–
	ρ_4	64.79	56.99	67.46	75.14	–	–
Balance-scale	ρ_1	5.95	6.07	5.66	5.76	5.4	–
	ρ_2	89.84	90.48	91.53	91.66	91.84	–
	ρ_3	10.16	9.52	8.47	8.34	8.16	–
	ρ_4	89.88	89.76	91.11	91.76	91.38	–
Soybean	ρ_1	7.53	5.01	3.61	3.08	2.71	–
	ρ_2	88.24	75.51	61.69	47.06	36.92	–
	ρ_3	11.76	24.49	38.31	52.94	63.08	–
	ρ_4	94.11	88.13	82.19	76.58	70.03	–
Splice	ρ_1	4.08	3.58	3.4	3.35	3.35	–
	ρ_2	9.3	10.47	10.54	10.36	10.82	–
	ρ_3	90.7	89.53	89.46	89.64	89.18	–
	ρ_4	19.2	23.63	25.85	26.06	27.12	–
Thyroid-allrep	ρ_1	1.6	1.24	1.11	1.05	1.09	–
	ρ_2	19.22	28.15	32.29	34.49	35.56	–
	ρ_3	80.78	71.85	67.71	65.51	64.44	–
	ρ_4	46.22	48.58	50.72	56.74	60	–
Thyroid-ann	ρ_1	5.18	5.47	5.6	5.73	5.77	–
	ρ_2	90.37	90.99	91.63	91.77	92.16	–
	ρ_3	9.63	9.01	8.37	8.23	7.84	–
	ρ_4	91.32	92.09	92.65	93.06	93.22	–
Car	ρ_1	31.19	33.19	33.30	–	–	–
	ρ_2	36.06	42.04	41.97	–	–	–
	ρ_3	63.94	57.96	58.03	–	–	–
	ρ_4	54.03	64.23	67.08	–	–	–
Nursery	ρ_1	8.61	14.43	12.01	13.82	11.94	11.71
	ρ_2	90.83	53.24	79.24	70.80	78.98	80.38
	ρ_3	9.17	46.76	20.76	29.20	21.02	19.62
	ρ_4	91.31	92.42	92.89	87.23	89.27	88.18
Page-blocks	ρ_1	18.09	19.49	19.73	–	–	–
	ρ_2	82.03	83.12	81.84	–	–	–
	ρ_3	17.97	16.88	18.16	–	–	–
	ρ_4	85.20	87.20	86.74	–	–	–
Optdigits	ρ_1	14.97	23.21	23.50	12.79	–	–
	ρ_2	97.74	39.41	42.59	36.10	–	–
	ρ_3	2.26	60.59	57.41	63.90	–	–
	ρ_4	98.31	83.96	72.98	77.72	–	–
Vehicle	ρ_1	16.87	41.67	42.61	37.70	–	–
	ρ_2	95.96	62.37	60.96	45.23	–	–
	ρ_3	4.04	37.63	39.04	54.77	–	–
	ρ_4	90.59	57.81	53.09	65.60	–	–

FIGURE 12. This is table11

TABLE XII THE COUNT OF WIN-DRAW-LOSE AMONG SFL WITH DIFFERENT λ VALUES, WHERE S_i DENOTES THE iTH DATA SUBSET. WILCOXON SIGNED-RANK TEST WITH THE LEVEL OF SIGNIFICANCE $\alpha = 0.05$ WAS EMPLOYED FOR COMPARING THE OVERALL RECALLS AFTER TRAINING WITH EACH DATA SUBSET.

	$\lambda \rightarrow$	0	0.25	0.5	0.75	1
Synthetic Type A	S_1	0-4-0	0-3-1	0-4-0	0-4-0	1-3-0
	S_2	0-3-1	0-2-2	3-1-0	2-2-0	0-2-2
	S_3	1-3-0	0-3-1	2-2-0	0-4-0	0-2-2
	S_4	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
	S_5	2-2-0	0-4-0	0-4-0	0-3-1	0-3-1
Synthetic Type B	S_1	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
	S_2	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
	S_3	1-3-0	1-3-0	2-2-0	1-2-1	0-0-4
	S_4	3-1-0	2-2-0	1-2-1	1-1-2	0-0-4
	S_5	2-2-0	1-3-0	1-3-0	1-2-1	0-0-4
Synthetic Type C	S_1	2-2-0	1-3-0	1-2-1	1-3-0	0-0-4
	S_2	1-0-3	2-2-0	2-2-0	2-2-0	0-0-4
	S_3	2-2-0	0-4-0	0-4-0	0-3-1	0-3-1
	S_4	1-3-0	1-3-0	1-3-0	0-0-4	1-3-0
Balance-scale	S_1	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
	S_2	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_3	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_4	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_5	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
Soybean	S_1	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_2	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_3	0-4-0	1-3-0	0-4-0	0-3-1	0-4-0
	S_4	0-4-0	0-4-0	0-4-0	0-3-1	1-3-0
	S_5	1-3-0	0-3-1	0-4-0	0-4-0	0-4-0
Splice	S_1	0-2-2	0-2-2	0-2-2	3-0-1	4-0-0
	S_2	0-2-2	0-3-1	1-2-1	0-3-1	4-0-0
	S_3	0-3-1	0-3-1	0-3-1	0-3-1	4-0-0
	S_4	0-3-1	0-1-3	1-2-1	1-2-1	4-0-0
	S_5	0-3-1	0-3-1	0-3-1	0-3-1	4-0-0
Thyroid-allrep	S_1	1-3-0	3-1-0	1-2-1	1-2-1	0-0-4
	S_2	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_3	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_4	0-3-1	0-3-1	0-2-2	3-1-0	1-3-0
	S_5	0-2-2	0-3-1	0-4-0	2-2-0	1-3-0
Thyroid-ann	S_1	0-3-1	0-3-1	0-4-0	0-3-1	3-1-0
	S_2	0-4-0	0-2-2	1-3-0	1-3-0	0-4-0
	S_3	1-1-2	0-3-1	3-1-0	2-2-0	0-1-3
	S_4	1-1-2	1-2-1	3-1-0	2-2-0	0-0-4
	S_5	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
Car	S_1	0-3-1	0-3-1	0-3-1	0-3-1	4-0-0
	S_2	2-2-0	2-2-0	1-1-2	0-0-4	1-3-0
	S_3	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
Nursery	S_1	3-1-0	3-1-0	2-0-2	0-1-3	0-1-3
	S_2	3-1-0	2-2-0	2-1-1	0-1-3	0-1-3
	S_3	3-1-0	3-1-0	2-0-2	1-0-3	0-0-4
	S_4	4-0-0	2-1-1	0-2-2	0-3-1	0-2-2
	S_5	2-2-0	3-1-0	2-1-1	1-0-3	0-0-4
	S_6	3-1-0	3-1-0	2-0-2	1-0-3	0-0-4
Page-blocks	S_1	1-3-0	1-3-0	1-3-0	1-3-0	0-0-4
	S_2	2-2-0	2-2-0	2-2-0	1-0-3	0-0-4
	S_3	2-2-0	1-3-0	2-2-0	1-1-2	0-0-4
Optdigits	S_1	0-4-0	0-4-0	0-4-0	0-4-0	0-4-0
	S_2	2-2-0	1-3-0	2-2-0	1-1-2	0-0-4
	S_3	3-1-0	3-1-0	2-0-2	1-0-3	0-0-4
	S_4	4-0-0	3-0-1	2-0-2	1-0-3	0-0-4
Vehicle	S_1	0-4-0	2-2-0	0-3-1	0-3-1	0-4-0
	S_2	2-2-0	2-2-0	1-1-2	0-0-4	1-3-0
	S_3	3-1-0	2-2-0	2-1-1	0-1-3	0-1-3
	S_4	2-2-0	3-1-0	1-2-1	1-1-2	0-0-4

FIGURE 13. This is table12

better on class 2 and class 4 when they first come up, it degrades much more on other classes. Therefore, it is not surprising that SFL gets the best overall recalls.

The experimental results indicate that the performance of Learn++.UDNC usually leans to majority classes. Even though it sometimes performs better on minority classes, the performance on other classes are usually degraded too much. On contrary, SFL can usually get more balanced performance on different classes and get better overall performance. This is because of the different processing methods of SFL and Learn++.UDNC for handling class imbalance problems. In SFL, class imbalance is considered when training the model. The method for training MLPs has been shown to be effective for class imbalance problems. In Learn++.UDNC, the training process did not consider class imbalance and a transfer function with consideration of class imbalance was applied to the outputs. The effectiveness of the method has not been well proved. Even in the results presented in [5], the performance on minority classes was much worse than that of majority classes. Therefore, it is not surprising that SFL can outperform Learn++.UDNC on most of the data sets.

4.3. **Computational time.** The computational time of SFL and Learn++.UDNC on all the data sets is presented in TABLE X. It can be observed from TABLE X that SFL usually takes less computational time than Learn++.UDNC. In the experiments, the structures of MLPs were the same for SFL and Learn++.UDNC and the stop criterion were also the same. However, more MLPs were trained for Learn++.UDNC for every new data subset. On the other hand, the training process of SFL usually meet the stop criterion earlier than that of Learn++.UDNC. Therefore, SFL is usually faster than Learn++.UDNC.

4.4. **Analyses about the components of SFL.** In SFL, two kinds of base classifiers, i.e., MLPs and NB, are employed to construct the ensemble. The results have shown that SFL is capable of outperforming the models with only MLPs and the models with only NB. To find out the reason, the differences of SFL and its components (MLPs and NB) and the influences of the differences are investigated in detail.

After every data subset is learned, four numbers are estimated on testing data set: the number of the examples that are correctly classified by only MLPs($\#_1$) or NB ($\#_2$), the number of the examples that are correctly classified by only MLPs or NB and correctly classified by SFL ($\#_3$). Then four ratios are estimated:

$$\begin{cases} \rho_1 = (\#_1 + \#_2)/\#_t \\ \rho_2 = \#_1/(\#_1 + \#_2) \\ \rho_3 = \#_2/(\#_1 + \#_2) \\ \rho_4 = \#_3/(\#_1 + \#_2) \end{cases} \tag{11}$$

where $\#_t$ is the number of examples in testing data set. The ratios are estimated for all the data sets and the average values over 30 executions are presented in TABLE XI. ρ_1 indecates the diversity (on making correct classification decisions) between MLPs and NB. ρ_4 indecates the benefits that SFL gets from the difference between MLPs and NB. It can be observed from TABLE XI that the values of ρ_4 are always closer to the larger one of rho_2 and rho_3 and sometimes exceed both of them. The observations partially show the reason that SFL always performs toward the better one of MLPs and NB and sometimes exceeds both of them.

4.5. **Analyses of parameters.** There are some parameters in SFL, including the number of MLPs, the number of hidden nodes in every MLP, the stop criterion for training MLPs and the coefficient λ in NCL. In our experimental studies, the number of MLPs and the number of hidden nodes in every MLP were set by experience. An independent execution was implemented for every data set to set the stop criterion to ensure the convergence of the training process. The coefficient λ in NCL is a parameter for controlling the diversities between the individuals in the ensemble (larger λ will lead to larger diversities). In the study of NCL[14], λ was suggested to be between 0 and 1. In our experimental studies, it was set to 0.5 for all the data sets. Since diversity is a very important issue for the success of ensemble learning methods [25], it is worthy to see the difference performance of SFL with different λ.

Extra executions of SFL with $\lambda = 0, 0.25, 0.75$ and 1 were conducted for all the used data sets. Wilcoxon signed-rank test with the level of significance $\alpha = 0.05$ was employed for comparing the overall recalls after training with each data subset. The results of every setting of λ were compared with the results of the other four settings of λ and the number of windraw-lose was counted and presented in TABLE XII. It can be observed from TABLE XII that λ affects the performance on most data sets. On some data sets, we can also observe the trend that the performance becomes better as λ decreases, such as Synthetic Type A, Synthetic Type B, Nursery, Page-blocks, Optdigits and Vehicle. In SFL, λ is not the only factor for encourage diversities. On one hand, the model built by NB may be quite different from the MLPs. On the other hand, in incremental learning, different MLPs may be trained with different data subsets, which will also result in diversities, especially when the data subsets are quite different. Therefore,large λ (such as 1) may emphasize too much to produce diversities so that the performance may be degraded.

5. **Conclusions and future work.** This paper investigates incremental learning in class imbalance situation. An ensemble-based method, i.e., SFL, which is a hybrid of MLPs and NB, was proposed. A group of impact weights (with the number of the classes as the length) was updated for every individual of the ensemble to indicate the confidence of the individual learning about the classes. The weights affect the outputs of the ensemble by weighted aver-age of all individuals outputs. The training of MLPs and NB considered class imbalance so that the ensemble can adapt the situation of class imbalance.

The experimental studies on 3 synthetic data sets and 10 real-world data sets have shown that the performance of SFL was better than that of a recently proposed approach for class imbalance incremental learning, i.e. Learn++.UDNC[9]. The experimental results have also shown that SFL can combine the advantages of both MLPs and NB to make a better model.

SFL has successfully combined MLPs and NB. The experimental studies have shown that combining additive models can make progress in incremental learning. However, this is just an ordinary trial. Other additive models, such as parame-ter estimation model might also help to improve SFL. This would be a direction of our future work.

REFERENCES

[1] A. Asuncion and D. Newman, Uci machine learning repository, 2007.

[2] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds and D. B. Rosen, Fuzzy artmap: A neural network architecture for incremental supervised learning of analog multidimensional maps, *IEEE Transactions on Neural Networks*, **3** (1992), 698–713.
[3] G. A. Carpenter, S. Grossberg and J. H. Reynolds, *ARTMAP: Supervised Real-Time Learning and Classification of Nonstationary Data by a Self-Organizing Neural Network*, Elsevier Science Ltd., 1991.
[4] N. V. Chawla, N. Japkowicz and A. Kotcz, Editorial: Special issue on learning from imbalanced data sets, *Acm Sigkdd Explorations Newsletter*, **6** (2004), 1–6.
[5] G. Ditzler, M. D. Muhlbaier and R. Polikar, Incremental learning of new classes in unbalanced datasets: Learn?+?+?.UDNC, *International Workshop on Multiple Classifier Systems, Multiple Classifier Systems,* (2010), 33–42.
[6] G. Ditzler, R. Polikar and N. Chawla, An incremental learning algorithm for non-stationary environments and class imbalance, In *International Conference on Pattern Recognition*, (2010), 2997–3000.
[7] Y. Freund and R. E. Schapire, A short introduction to boosting, *Journal of Japanese Society for Artificial Intelligence*, **14** (1999), 771–780.
[8] L. Fu, H.-H. Hsu and J. C. Principe, Incremental backpropagation learning networks, *IEEE Transactions on Neural Networks*, **7** (1996), 757–761.
[9] H. He and E. A. Garcia, Learning from imbalanced data, *IEEE Transactions on Knowledge and Data Engineering*, **21** (2009), 1263–1284.
[10] H. Inoue and H. Narihisa, Self-organizing neural grove and its applications, In *IEEE International Joint Conference on Neural Networks*, **2** (2005), 1205–1210.
[11] N. Japkowicz and S. Stephen, *The Class Imbalance Problem: A Systematic Study*, IOS Press, 2002.
[12] N. Kasabov, Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning, *IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society*, **31** (2001), 902–918.
[13] M. Lin, K. Tang and X. Yao, Dynamic sampling approach to training neural networks for multiclass imbalance classification, *IEEE Transactions on Neural Networks and Learning Systems*, **24** (2013), 647–660.
[14] Y. Liu and X. Yao, Simultaneous training of negatively correlated neural networks in an ensemble, *IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society*, **29** (1999), 716–725.
[15] F. L. Minku, H. Inoue and X. Yao, Negative correlation in incremental learning, *Natural Computing*, **8** (2009), 289–320.
[16] M. Muhlbaier, A. Topalis and R. Polikar, Incremental learning from unbalanced data, In *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, IEEE, **2** (2004), 1057–1062.
[17] M. Muhlbaier, A. Topalis and R. Polikar, Learn++.mt: A new approach to incremental learning, *Lecture Notes in Computer Science*, **3077** (2004), 52–61.
[18] M. D. Muhlbaier, A. Topalis and R. Polikar, Learn ++.nc: combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes, *IEEE Transactions on Neural Networks*, **20** (2009), p152.
[19] S Ozawa, S. Pang and N. Kasabov, Incremental learning of chunk data for online pattern classification systems, *IEEE Trans Neural Netw*, **19** (2008), 1061–1074.
[20] R. Polikar, J. Byorick, S. Krause and A. Marino, Learn++: A classifier independent incremental learning algorithm for supervised neural networks, In *International Joint Conference on Neural Networks*, (2002), 1742–1747.
[21] R. Polikar, L. Upda, S. S. Upda and V. Honavar, Learn++: an incremental learning algorithm for supervised neural networks, *IEEE Transactions on Systems Man & Cybernetics Part C*, **31** (2001), 497–508.
[22] M. Salganicoff, Tolerating concept and sampling shift in lazy learning using prediction error context switching, *Artificial Intelligence Review*, **11** (1997), 133–155.
[23] M. C. Su, J. Lee and K. L. Hsieh, A new artmap-based neural network for incremental learning, *Neurocomputing*, **69** (2006), 2284–2300.
[24] Y. Sun, M. S. Kamel and Y. Wang, Boosting for learning multiple classes with imbalanced class distribution, In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, IEEE, (2006), 592–602.

[25] E. K. Tang, P. N. Suganthan and X. Yao, An analysis of diversity measures, *Machine Learning*, **65** (2006), 247–271.
[26] K. Tang, M. Lin, F. L. Minku and X. Yao, Selective negative correlation learning approach to incremental learning, *Neurocomputing*, **72** (2009), 2796–2805.
[27] W. X. Wen, H. Liu and A. Jennings, Self-generating neural networks, In *International Joint Conference on Neural Networks*, **4** (2002), 850–855.
[28] G. Widmer and M. Kubat, Effective learning in dynamic environments by explicit context tracking, In *Machine learning: ECML-93*, Springer, **667** (1993), 227–243.
[29] J. R. Williamson, Gaussian artmap: A neural network for fast incremental learning of noisy multidimensional maps, *Neural Networks*, **9** (1996), 881–897.

E-mail address: sunnyboy@mail.ustc.edu.cn
E-mail address: ketang@ustc.edu.cn