Chapter 10
Real-Time Approaches to Computational Economics: Self Adaptive Economic Systems

Abstract This chapter examines modelling of financial movement direction with Learn++ by forecasting the daily movement direction of the Dow Jones. The Learn++ approach is implemented using a multi-layer perceptron as a weak-learner, where this weak-learner is improved by making use of the Learn++ algorithm. In addition, the Learn++ algorithm introduces the concept of on-line incremental learning, which means that the proposed framework is able to adapt to new data.

10.1 Introduction

This chapter assumes that a complete model is the one that is able to continuously self-adapt to the changing environment. In this chapter, an on-line incremental algorithm that classifies the direction of movement of the stock market is described (Lunga and Marwala 2006a). One very important component of the economic system is the financial market. The financial market is a complex, evolving, and non-linear dynamic system. In order to increase the wealth of investors it is vital to be able to forecast the direction of the financial markets. The field of financial forecasting is manifested by data intensity, noise, non-stationarity, unstructured nature, high degree of uncertainty, and hidden relationships (Carpenter et al. 1992; Lunga and Marwala 2006a, b). Various aspects interact in finance, and these include social forces, political developments, overall economic conditions, and traders' expectations. Consequently, predicting market price movements is a difficult undertaking. Movements of market prices are not entirely random and they behave in a highly non-linear and dynamic manner. The standard random walk assumption of future prices is a different manifestation of randomness that hides a noisy non-linear process (McNelis 2005).

T. Marwala, *Economic Modeling Using Artificial Intelligence Methods*, Advanced Information and Knowledge Processing, DOI 10.1007/978-1-4471-5010-7_10,

Incremental learning is a possible solution to such situations and is defined as the process of extracting new information from the data without losing prior knowledge from an additional dataset that later becomes available (Lunga and Marwala 2006a). A number of definitions and interpretations of incremental learning can be found in the literature, including on-line learning (Freund and Schapire 1997; Lunga and Marwala 2006b), re-learning of previously misclassified instances, and growing and pruning of classifier architectures (Bishop 1995). An algorithm possesses incremental learning capabilities if it meets the following criteria (Lunga and Marwala 2006b):

- Capability to attain further knowledge when new data are introduced.
- Capability to remember previously learned information about the data.
- Capability to learn new classes of data if introduced by new data.

Some applications of on-line classification problems have been reported recently (Polikar et al. 2002, 2004; Polikar 2000; Vilakazi et al. 2006; Vilakazi 2007). In many situations, the extent of accuracy and the acceptability of certain classifications are measured by the error of misclassified instances. Learn++ has mostly been applied to classification problems and the choice of Learn++ algorithm can boost a weak-learner to classify stock closing values with minimum error and reduced training time (Lunga and Marwala 2006b). For financial markets, forecasting methods based on minimizing forecasting error may not be sufficient. Trade driven by a certain forecast with a small forecast error may not be as profitable as trade guided by an accurate prediction of the direction of movement. This chapter discusses the ensemble systems, introduces the basic theory of incremental learning and the Learn++ algorithm, and applies these to financial markets.

10.2 Incremental Learning

An incremental learning algorithm is defined as an algorithm that learns new information from unseen data, without requiring access to previously observed data (Polikar et al. 2002, 2004; Polikar 2000). The algorithm is capable of learning newly available information from the data and to recall the knowledge from the previously observed data. Furthermore, the algorithm is capable of learning new classes that are introduced by subsequent data. This kind of learning algorithm is called an on-line learning procedure. Learning new information without accessing previously used data invokes the 'stability-plasticity dilemma' (Carpenter et al. 1992). A completely stable classifier retains the knowledge from previously learned data but fails to learn new information while a completely plastic classifier learns new data but forgets prior knowledge. The problem with neural network techniques is that they are stable classifiers and cannot learn new information after they have been trained. Different measures have been applied to capacitate neural networks with incremental learning capability. One technique of learning new information

from supplementary data entails eliminating the trained classifier and training a new classifier using accumulated data. Other approaches such as pruning of networks or controlled alteration of classifier weight or growing of classifier architectures are known as incremental learning techniques and they change classifier weights using only the misclassified instances. These techniques are capable of learning new information, nevertheless, they suffer from 'forgetting' and necessitate access to old data. One method evaluates the current performance of the classifier architecture. If the present architecture does not adequately characterize the decision boundaries being learned, new decision clusters are generated in response to new pattern. Additionally, this method does not involve access to old data and can accommodate new classes. Nonetheless, the central inadequacies of this method are: cluster proliferation and sensitivity to selection of algorithm parameters. In this chapter, Learn++ is applied for on-line prediction of stock movement direction.

10.3 Ensemble Methods

The on-line learning technique implemented in this chapter is based on ensemble learning (Hansen and Salamon 1990; Jordan and Jacobs 1994; Kuncheva et al. 2001). *Ensemble learning* is a method where multiple models, such as classifiers, are deliberately created and combined to solve a particular problem (Rogova 1994; Polikar 2006; Marwala 2012). These techniques combine an ensemble of usually weak classifiers to exploit the so-called instability of the weak classifier (Polikar 2006). A tactical mixture of these classifiers eradicates the individual errors, creating a strong classifier. This makes the classifiers build adequately different decision boundaries for negligible changes in their training parameters and, as a result, each classifier makes different errors on any given instance. Ensemble systems have enticed a great deal of attention over the last decade due to their empirical success over single classifier systems on a variety of applications (Hulley and Marwala 2007; Marwala 2009).

Hannah and Dunson (2012) successfully applied an ensemble method to geometric programming based circuit design whereas Tong et al. (2012) successfully applied an ensemble of Kalman filters for approximating a heterogeneous conductivity field by integrating transient solute transport data. Austin et al. (2012) successfully applied ensemble methods for forecasting mortality in patients with cardiovascular disease whereas Halawani and Ahmad (2012) successfully applied ensemble methods to predict Parkinson disease and Ebrahimpour et al. (2012) successfully applied ensemble method to detect epileptic seizure.

In Sect. 10.3 ensemble learning methods are described: bagging, stacking and adaptive boosting (Marwala 2012). Particularly, the Adaptive Boosting technique is described because it was the basis for the creation of the Learn++ procedure, which is the on-line routine, implemented in this chapter (Polikar 2006).

10.3.1 Bagging

Bagging is a technique which is premised on the combination of models fitted to randomly chosen samples of a training data set to decrease the variance of the prediction model (Efron 1979; Breiman 1996; Marwala 2012). Bagging essentially necessitates randomly choosing a subset of the training data and applying this subset to train a model and repeating this process. Subsequently, all trained models are combined with equal weights to form an ensemble.

Louzada and Ara (2012) successfully applied bagging for fraud detection tool whereas Ghimire et al. (2012) successfully applied bagging for land-cover classification in Massachusetts. Syarif et al. (2012) successfully applied bagging to intrusion detection whereas Zhang et al. (2012) successfully applied bagging for high resolution range profile recognition for polarization radar.

10.3.2 Stacking

A model can be chosen from a set of models by comparing these models using data that was not used to train the models (Polikar 2006; Marwala 2012). This prior belief can also be applied to select a model amongst a set of models, based on a single data set by using a method called *cross-validation* (Bishop 1995; Marwala 2012). This is accomplished by dividing the data into a *training* data set, which is used to train the models, and a *testing* data set which is used to test the trained model. Stacking takes advantage of this prior belief by using the performance from the test data to combine the models instead of choosing among them the best performing model when tested on the testing data set (Wolpert 1992).

Sulzmann and Fürnkranz (2011) successfully applied stacking to compress an ensemble of rule sets into a single classifier whereas Chen and Wong (2011) successfully applied ant colony optimization method to optimize stacking ensemble. Lienemann et al. (2009) successfully applied stacking in metabonomic applications.

10.3.3 Adaptive Boosting (AdaBoost)

Boosting is a technique that incrementally generates an ensemble by training each new model with data that the previously trained model misclassified. Then the ensemble, which is a combination of all trained models, is used for prediction. Adaptive Boosting is an extension of boosting to multi-class problems (Freund and Schapire 1997; Schapire et al. 1998; Marwala 2012). There are many types of Adaptive Boosting, for instance AdaBoost.M1, where each classifier is assigned

a weighted error of no more than ½ and AdaBoost.M2 with weak classifiers with a weighted error of less than ½. Tang et al. (2008) applied successfully Adaptive Boosting technique for analog circuit fault diagnosis whereas Li and Shen (2008) successfully applied Adaptive Boosting method for image processing. Nock et al. (2012) successfully applied boosting to classify natural scenes whereas Xia et al. (2012) successfully applied boosting for image retrieval. La et al. (2012) successfully applied boosting for text classification.

For AdaBoost.M1, samples are drawn from a distribution D that is updated in such a way that successive classifiers concentrate on difficult cases. This is achieved by adjusting D in such a way that the earlier, misclassified cases are likely to be present in the following sample. The classifiers are then combined through weighted majority voting. The distribution begins as a uniform distribution so that all cases have equal probability can be drawn into the first data subset S_1.

As described by Polikar (2006), at each iteration t, a new training data subset is sampled, and a weak classifier is trained to create a hypothesis h_t. The error given by this hypothesis with regards to the current distribution is estimated as the sum of distribution weights of the cases misclassified by h_t. AdaBoost.M1 requires that this error is less than ½, and if this requirement is violated then the procedure terminates. The normalized error β_t is then calculated so that the error that is in the [0 0.5] interval is normalized into the [0 1] interval. The transformed error is implemented in the distribution update rule, where $D_t(i)$ is decreased by a factor of $\beta_t, 0 < \beta_t < 1$, if x_i is correctly classified by h_t, or else it is left unaltered. When the distribution is normalized so that $D_{t+1}(i)$ is a proper distribution, the weights of those instances that are misclassified are increased. This update rule guarantees that the weights of all instances are correctly classified and the weights of all misclassified instances add up to ½. The requirement for the training error of the base classifier to be less than ½ forces the procedure to correct the error committed by the previous base model. When the training process is complete, the test data are classified by this ensemble of T classifiers, by applying a weighted majority voting procedure where each classifier obtains a voting weight that is inversely proportional to its normalized error (Polikar 2006). The weighted majority voting then selects the class ω allocated the majority vote of all classifiers. The procedure for Adaptive Boosting is shown in Algorithm 10.1 (Polikar 2006).

As described by Polikar (2006), the theoretical analysis of the Adaptive Boosting technique shows that the ensemble training error E is bounded above by:

$$E < 2^T \prod_{t=1}^{T} \sqrt{\varepsilon_t (1 - \varepsilon_t)} \tag{10.1}$$

The $\varepsilon_t < 1/2$ ensemble error E is reduced when new classifiers are added. The Adaptive Boosting method is not prone to over-fitting and this is explained by the margin theory (Schapire 1990; Polikar 2006).

Algorithm 10.1 The AdaBoost Algorithm.M1

Input:

- Input $X = \{x_1, x_2, \ldots, x_n\}$ and output $Y = \{y_1, y_2, \ldots, y_n\}$
- Weak-learner algorithm
- Number of classifiers T and distribution $D_1(i) = 1/n; i = 1, \ldots, n$

For $t = 1,2,\ldots,T$;

1. Sample a training subset S_t with a distribution D_t
2. Train *Weak-learner* with S_t and create hypothesis $h_t : X \to Y$
3. Estimate the error of $h_t : \varepsilon_t = \sum_{i=1}^{n} I\left[h_t(x_i) \neq y_i\right] \cdot D_t(i) = \sum_{t:h_t(\mathbf{x}_i)\neq y_i} D_t(i)$
4. If $\varepsilon_t > \frac{1}{2}$ terminate
5. Estimate the normalized error $\beta_t = \varepsilon_t / (1 - \varepsilon_t) \Rightarrow 0 \leq \beta_t \leq 1$
6. Update the distribution D_t: $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \beta_t, if\ h_t(x_i)\ y_i \\ 1, otherwise \end{cases}$ where Z_t is the normalization constant so that D_{t+1} becomes a proper distribution function.

Test using majority voting given an unlabeled example z as follows:

- Count the total vote from the classifiers $V_j = \sum_{t:h_t(z)} \log(1/\beta_t)$ $j = 1, \ldots, C$
- Select the class that receives the highest number of votes as the final classification.

10.4 The Real-Time Method

Real-time learning is suitable for modelling dynamically time-varying systems where the characteristics of the environment in which the system is operating changes with time. It is also suitable when the data set existing is inadequate and does not entirely describe the system and, therefore, this approach incorporates new conditions that may be presented by newly acquired data.

A real-time computational economics model must have incremental learning competency if it is to be applied for automatic and continuous real-time prediction. The basis of real-time learning is incremental learning, which has been studied by many researchers (Higgins and Goodman 1991; Fu et al. 1996; Yamaguchi et al. 1999; Carpenter et al. 1992; Marwala 2012). The difficulty with real-time learning

is the tendency of a real-time learner to forget the information learned during the initial stages of the learning process (McCloskey and Cohen 1989). The real-time learning technique adopted for this chapter is Learn++ and was proposed by Polikar et al. (2004).

Vilakazi and Marwala (2007a) applied the real-time incremental learning technique for monitoring the condition of high voltage bushings. Two incremental learning techniques were applied to the problem of condition monitoring. The first technique used was the incremental learning capability of the Fuzzy ARTMAP (FAM), and they investigated whether the ensemble approach can improve the performance of the FAM. The second technique applied was Learn++ that implemented an ensemble of multi-layer perceptron classifiers. Both methods performed well when tested for transformer bushing condition monitoring.

Mohamed et al. (2007) applied incremental learning for the classification of protein sequences. They used the fuzzy ARTMAP as an alternative machine learning system with the ability to incrementally learn new data as it becomes available. The fuzzy ARTMAP was seen to be comparable to many other machine learning systems. The application of an evolutionary strategy in the selection and combination of individual classifiers into an ensemble system, coupled with the incremental learning capability of the fuzzy ARTMAP was shown to be suitable as a pattern classifier. Their algorithm was tested using the data from the G-Coupled Protein Receptors Database and it demonstrated a good accuracy of 83 %.

Mohamed et al. (2006) applied fuzzy ARTMAP to multi-class protein sequence classification. They presented a classification system that used pattern recognition method to produce a numerical vector representation of a protein sequence and then classified the sequence into a number of given classes. They applied fuzzy ARTMAP classifiers and showed that, when coupled with a genetic algorithm based feature subset selection, the system could classify protein sequences with an accuracy of 93 %. This accuracy was then compared to other classification techniques and it was shown that the fuzzy ARTMAP was most suitable because of its high accuracy, quick training times and ability to learn incrementally.

Perez et al. (2010) applied a population-based, incremental learning approach to microarray gene expression feature selection. They evaluated the usefulness of the Population-Based Incremental Learning (PBIL) procedure on identifying a class differentiating gene set for sample classification. PBIL was based on iteratively evolving the genome of a search population by updating a probability vector, guided by the extent of class-separability demonstrated by a combination of features. The PBIL was then compared to standard Genetic Algorithm (GA) and an Analysis of Variance (ANOVA) method. The procedures were tested on a publicly available three-class leukemia microarray data set ($n = 72$). After running 30 repeats of both GA and PBIL, the PBIL could identify an average feature-space separability of 97.04 % while the GA achieved an average class-separability of 96.39 %. The PBIL also found smaller feature-spaces than GA, (PBIL – 326 genes and GA – 2,652) thus excluding a large percentage of redundant features. It also, on average, outperformed the ANOVA approach for $n = 2{,}652$ (91.62 %),

$q < 0.05$ (94.44 %), $q < 0.01$ (93.06 %) and $q < 0.005$ (95.83 %). The best PBIL run (98.61 %) even outperformed ANOVA for $n = 326$ and $q < 0.001$ (both 97.22 %). PBIL's performance was credited to its ability to direct the search, not only towards the optimal solution, but also away from the worst.

Hulley and Marwala (2007) applied GA-based incremental learning for optimal weight and classifier selection. They then compared Learn++, which is an incremental learning algorithm to the new Incremental Learning Using Genetic Algorithm (ILUGA). Learn++ demonstrated good incremental learning capabilities on benchmark datasets on which the new ILUGA technique was tested. ILUGA showed good incremental learning ability using only a few classifiers and did not suffer from catastrophic forgetting. The results obtained for ILUGA on the Optical Character Recognition (OCR) and Wine datasets were good, with an overall accuracy of 93 and 94 %, respectively, showing a 4 % improvement over Learn++. MT for the difficult multi-class OCR dataset.

Lunga and Marwala (2006a) applied a time series analysis using fractal theory and real-time ensemble classifiers to model the stock market. The fractal analysis was implemented as a concept to identify the degree of persistence and self-similarity within the stock market data. This concept was carried out using the Rescaled range analysis (R/S) technique. The R/S analysis outcome was then applied to a real-time incremental algorithm (Learn++) that was built to classify the direction of movement of the stock market. The use of fractal geometry in this study provided a way of determining, quantitatively, the extent to which the time series data could be predicted. In an extensive test, it was demonstrated that the R/S analysis provided a very sensitive technique to reveal hidden long runs and short run memory trends within the sample data. A time series data that was measured to be persistent was used to train the neural network. The results from the Learn++ algorithm showed a very high level of confidence for the neural network to classify sample data accurately.

Lunga and Marwala (2006b) applied incremental learning for the real-time forecasting of stock market movement direction. In particular, they presented a specific application of the Learn++ algorithm, and investigated the predictability of financial movement direction with Learn++ by forecasting the daily movement direction of the Dow Jones. The framework was implemented using the multi-layer perceptron (MLP) as a weak-learner. First, a weak learning algorithm, which tried to learn a class concept with a single input perceptron, was established. The Learn++ algorithm was then applied to improve the weak MLP learning capacity and thus introduced the concept of incremental real-time learning. The presented framework could adapt as new data were introduced and could classify the data well. This chapter is based on this study by Lunga and Marwala (2006b).

Vilakazi and Marwala (2007b) applied incremental learning to bushing condition monitoring. They presented a technique for bushing fault condition monitoring using the fuzzy ARTMAP. The fuzzy ARTMAP was introduced for bushing condition monitoring because it can incrementally learn information as it becomes available. An ensemble of classifiers was used to improve the classification accuracy of the system. The test results showed that the fuzzy ARTMAP ensemble gave an

accuracy of 98.5 %. In addition, the results showed that the fuzzy ARTMAP could update its knowledge in an incremental fashion without forgetting the previously learned information.

Nelwamondo and Marwala (2007) successfully applied a technique for handling missing data from heteroskedasticity and non-stationary data. They presented a computational intelligence approach for predicting missing data in the presence of concept drift using an ensemble of multi-layer feed-forward neural networks. Six instances prior to the occurrence of missing data were used to approximate the missing values. The algorithm was applied to a simulated time series data set that resembled non-stationary data from a sensor. Results showed that the prediction of missing data in a non-stationary time series data was possible but was still a challenge. For one test, up to 78 % of the data could be predicted within a 10 % tolerance range of accuracy.

Khreich et al. (2012) conducted a survey of techniques for incremental learning of hidden Markov model parameters while Tscherepanow et al. (2011) applied hierarchical adaptive resonance theory network for the stable incremental learning of topological structures and associations from noisy data. Bouchachia (2011) studied incremental learning with multi-level adaptation. The author examined self-adaptation of classification systems which were natural adaptation of the base learners to change in the environment, contributive adaptation when combining the base learners in an ensemble, and structural adaptation of the combination as a form of dynamic ensemble. The author observed that this technique was able to deal with dynamic change in the presence of various types of data drift.

Martínez-Rego et al. (2011) proposed a robust incremental learning technique for non-stationary environments. They proposed a method, for single-layer neural networks, with a forgetting function in an incremental on-line learning procedure. The forgetting function offered a monotonically increasing significance to new data. Owing to the mixture of incremental learning and increasing significance assignment the network forgot quickly in the presence of changes while retaining a stable behavior when the context was stationary. The performance of the technique was tested over numerous regression and classification problems and the results were compared with those of previous works. The proposed procedure revealed high adaptation to changes while maintaining a low consumption of computational resources.

Yang et al. (2011) proposed an extreme and incremental learning based single-hidden-layer regularization ridgelet network which applied the ridgelet function as the activation function in a feed-forward neural network. The results showed that the method demonstrated incremental learning capability.

Topalov et al. (2011) successfully applied a neuro-fuzzy control of antilock braking system using a sliding mode incremental learning procedure. An incremental learning procedure was applied to update the parameters of the neuro-fuzzy controller. The application of this on the control of anti-lock breaking system model gave good results.

Folly (2011) proposed a method to optimally tune the parameters of power system stabilizers for a multi-machine power system using the Population-based

incremental learning (PBIL) procedure. The PBIL procedure is a method that combines features of genetic algorithms and competitive learning-based on artificial neural networks. The results showed that PBIL based power system stabilizers performed better than genetic algorithm based power system stabilizers over a range of operating conditions considered.

Other successful implementations of incremental learning techniques include its use in anomaly detection (Khreich et al. 2009), in human robot interaction (Okada et al. 2009), for real-time handwriting recognition (Almaksour and Anquetil 2009), for predicting human movement in a vehicle motion (Vasquez et al. 2009), in visual learning (Huang et al. 2009), in nuclear transient identification (Baraldi et al. 2011), in object detection and pose classification (Tangruamsub et al. 2012), in classification of Alzheimer's disease (Cho et al. 2012), in face recognition (Lu et al. 2012) as well as in speech recognition (Li et al. 2012).

10.4.1 Learn++ Incremental Learning Method

Learn++ is an incremental learning procedure that was proposed by Polikar and co-workers (Polikar et al. 2002, 2004; Muhlbaier et al. 2004; Erdem et al. 2005; Polikar 2006; Marwala 2012). It is based on adaptive boosting procedure and applies multiple classifiers to capacitate the system to learn incrementally. The procedure operates on the notion of using many classifiers that are weak-learners to give a good overall classification. The weak-learners are trained on a separate subset of the training data and then the classifiers are combined using a weighted majority vote. The weights for the weighted majority vote are selected using the performance of the classifiers on the entire training dataset.

Each classifier is trained using a training subset that is sampled in accordance to a stated distribution. The classifiers are trained using a weak-learner approach. The condition for the weak-learner procedure is that it must give a classification rate of less than 50 % firstly (Polikar et al. 2002). For each database D_k that contains training series, S, where S contains learning examples and their equivalent classes, Learn++ starts by setting the weights vector, w, according to a specified distribution D_T, where T is the number of hypothesis. Firstly the weights are set to be uniform giving equal probability for all cases chosen for the first training subset and the distribution is then given by (Polikar et al. 2002; Marwala 2012):

$$D = \frac{1}{m} \tag{10.2}$$

Here, m is the number of training examples in S. The training data are then distributed into training subset TR and testing subset TE to confirm the weak-learner capability. The distribution is then applied to choose the training subset TR and testing subset TE from S_k. After training and testing subsets have been chosen, then

the weak-learner procedure is applied. The weak-learner is trained using subset *TR*. A hypothesis, h_t, attained from a weak-learner is tested using both the training and testing subsets to achieve an error (Polikar et al. 2002; Marwala 2012):

$$\varepsilon_t = \sum_{t:h_i(x_i)\neq y_i} D_t(i) \tag{10.3}$$

The error is required to be less than 0.5; a normalized error is computed using (Polikar et al. 2002; Marwala 2012):

$$B_t = \frac{\varepsilon_t}{1-\varepsilon_t} \tag{10.4}$$

If the error is greater than 0.5, the hypothesis is rejected and the new training and testing subsets are chosen according to a distribution D_T and another hypothesis is estimated. All classifiers created are then combined using weighted majority voting to obtain a combined hypothesis, H_t (Polikar et al. 2002; Marwala 2012):

$$H_t = \arg\max_{y\in Y} \sum_{t:h_t(x)=y} \log\left(\frac{1}{\beta_t}\right) \tag{10.5}$$

The weighted majority voting offers higher voting weights to a hypothesis that performs well on the training and testing data subsets. The error of the composite hypothesis is calculated as follows (Polikar et al. 2002; Marwala 2012):

$$E_t = \sum_{t:H_i(x_i)\neq y_i} D_t(i) \tag{10.6}$$

If the error is greater than 0.5, the current hypothesis is rejected and the new training and testing data are chosen according to a distribution D_T. Or else, if the error is less than 0.5, then the normalized error of the composite hypothesis is calculated as follows (Polikar et al. 2002; Marwala 2012):

$$B_t = \frac{E_t}{1-E_t} \tag{10.7}$$

The error is applied in the distribution update rule, where the weights of the correctly classified cases are reduced, accordingly increasing the weights of the misclassified instances. This confirms that the cases that were misclassified by the current hypothesis have a higher probability of being chosen for the succeeding training set. The distribution update rule is given by the following equation (Polikar et al. 2002; Marwala 2012):

$$w_{t+1} = w_t(i) \times B_t^{1-[|H_t(x_i)\neq y_i|]} \tag{10.8}$$

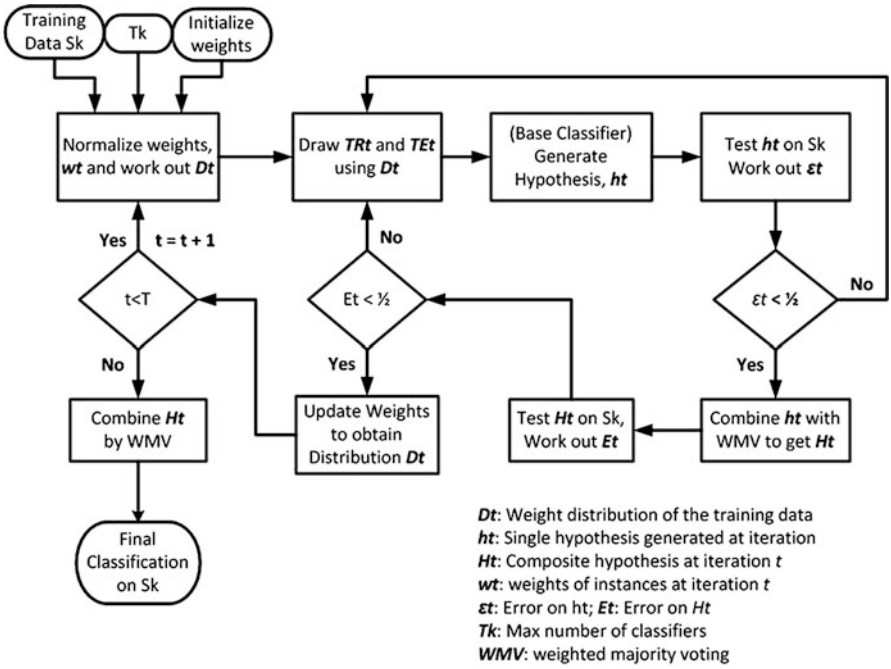

Fig. 10.1 Learn++ algorithm

After the T hypothesis has been generated for each database, the final hypothesis is calculated by combining the hypotheses using weighted majority voting as described by the following equation (Polikar et al. 2002; Marwala 2012):

$$H_t = \arg\max_{y \in Y} \sum_{k=1}^{K} \sum_{t:H_t(x)=y} \log\left(\frac{1}{\beta_t}\right) \tag{10.9}$$

The Learn++ algorithm is represented diagrammatically in Fig. 10.1 (Polikar 2006; Marwala 2012).

10.4.2 Confidence Measurement

To approximate the confidence of the Learn++ procedure, a majority of hypotheses agreeing on given instances is an indicator of confidence on the decision proposed. If it is assumed that a total of T hypotheses are generated in k training sessions for a C-class problem, then for any given example, the final classification class, the total vote class c received is given by (Muhlbaier et al. 2004; Marwala 2012):

Table 10.1 Confidence estimation representation (Lunga and Marwala 2006b)

Confidence range (%)	Confidence level
$0.9 \leq \lambda c \leq 1$	Very high (VH)
$0.8 \leq \lambda c \leq 0.8$	High (H)
$0.7 \leq \lambda c \leq 0.8$	Medium (M)
$0.6 \leq \lambda c \leq 0.7$	Low (l)
$0 \leq \lambda c \leq 0.6$	Very low (VL)

$$\zeta_c = \sum_{t:h_t(x)=c} \Psi_t \tag{10.10}$$

where Ψ_t denotes the voting weights of the t^{th}, hypothesis h_t.

Normalizing the votes received by each class can be performed as follows (Muhlbaier et al. 2004; Marwala 2012):

$$\lambda_c = \frac{\zeta_c}{\sum_{c=1}^{C} \zeta_c} \tag{10.11}$$

Here, λ_c can be interpreted as a measure of confidence on a scale of 0–1 and this representation is shown in Table 10.1 (Lunga and Marwala 2006b). A high value of λ_c shows high confidence in the decision and conversely, a low value of λ_c shows low confidence in the decision. It should be noted that the λ_c value does not represent the accuracy of the results, but the confidence of the system in its own decision.

10.4.3 Multi-layer Perceptron

In this chapter we use the multi-layer perceptron neural network to create a weak-learner. The multi-layered perceptrons have been successfully used to model complex systems (Marwala 2007), missing data estimation (Marwala 2009), inter-state conflict modelling (Marwala and Lagazio 2011) and condition monitoring (Marwala 2012). Each connection between inputs and neurons is weighted by adjustable weight parameters. Furthermore, each neuron has an adjustable bias weight parameter which is represented by a connection from a constant input $x_0 = 1$ and $z_0 = 1$ for the hidden neurons and the output neuron, respectively. This group of two-layer multi-layer perceptron models is capable of estimating any continuous function with arbitrary accuracy, providing the number of hidden neurons is appropriately large (Bishop 1995).

The advantage of the multi-layer perceptron network is the interconnected cross-coupling that occurs between the input variables and the hidden nodes, and the hidden nodes and the output variables. If we assume that x is the input to the

multi-layer perceptron and y is the output of the MLP, a mapping function between the input and the output may be written as follows (Bishop 1995):

$$y = f_{\text{output}}\left(\sum_{j=1}^{M} w_j\, f_{\text{hidden}}\left(\sum_{i=0}^{N} w_{ij}\, x_i\right) + w_0\right) \tag{10.12}$$

where N is the number of input units, M is the number of hidden neurons, x_i is the ith input unit, w_{ij} is the weight parameter between input i and hidden neuron j and w_j is the weight parameter between hidden neuron j and the output neuron. The activation function $f_{\text{output}}(\cdot)$ is sigmoid and can be written as follows (Bishop 1995):

$$f_{output}(a) = \frac{1}{1 + e^{-a}} \tag{10.13}$$

For classification problems, the sigmoid function is ideal (Bishop 1995). The activation function $f_{\text{hidden}}(\cdot)$ is a hyperbolic tangent can be written as follows (Bishop 1995):

$$f_{hidden}(a) = \tanh(a) \tag{10.14}$$

The neural network model in Eq. 10.12 is trained using the scaled conjugate gradient method, which is described in Bishop (1995).

10.5 Experimental Investigation

This analysis examines the daily changes of the Dow Jones Index. The Dow Jones averages are particular in that they are price weighted rather than market capitalization weighted. Their component weightings are consequently impacted only by changes in the stock prices, in contrast with other indexes' weightings that are impacted by both price changes and changes in the number of shares outstanding (Leung et al. 2000). When the averages were originally generated, their values were computed by merely totalling up the constituent stock prices and dividing by the number of constituents. Altering the divisor was started to isolate the consequences of stock separations and other corporate activities.

The Dow Jones Industrial Average measures the composite price performance of over 30 highly capitalized stocks trading on the New York Stock Exchange (NYSE), representing a broad cross-section of industries in the USA. Trading in the index has gained unparalleled reputation in foremost financial markets around the world. The increasing diversity of financial instruments associated to the Dow Jones Index has expanded the dimension of global investment prospect for both individual and institutional investors. There are two basic explanations for the success of these

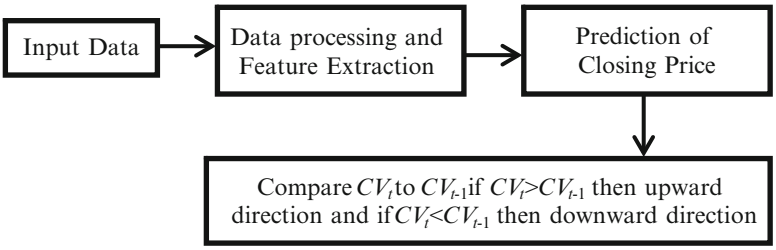

Fig. 10.2 Proposed model for real-time stock forecasting

index trading instruments. The first reason is that they afford an effective means for investors to hedge against potential market risks. The second reason is that they generate new profit making prospects for market investors. Consequently, it has deep consequences and importance for researchers and practitioners to correctly forecast the direction of the movement of stock prices.

Previous research has investigated the cross-sectional relationship between stock index and macroeconomic variables. Macroeconomic input variables which are normally implemented for forecasting include term structure of interest rates, short-term interest rate, long-term interest rate, consumer price index, industrial production, government consumption, private consumption and gross domestic product. In this chapter, the closing values of the index were selected as inputs.

A one step forward prediction of the index was performed on a daily basis. The output of this prediction model was used as input to the Learn++ algorithm for classification into the correct category that would give an indication of whether the predicted index value is 1 (indicating a positive increase in next day's predicted closing value compared to the previous day's closing value) or a predicted closing value of −1, indicating a decrease in next day's predicted closing value compared to the previous day's closing value. Figure 10.2 shows the conceptual model of all processes needed for this study (Lunga and Marwala 2006a). The first prediction model can be written as (Lunga and Marwala 2006a):

$$CV_t = F\left(cv_{t-1}, cv_{t-1}, cv_{t-1}, cv_{t-1}\right) \tag{10.15}$$

where CV_t is the predicted close value at time t, cv_{t-1} indicates the close value at day i, where $i = 1,2,3$, $t-1$. The second model takes the output of the first model as its input in predicting the direction of movement for the index. The classification prediction stage can be represented as (Lunga and Marwala 2006a):

$$Direction_t = F\left(CV_t\right) \tag{10.16}$$

where CV_t is the first model prediction of the fifth day stock closing value when given the raw data at time $t-1$ to $t-4$ respectively. *Directiont* is a categorical

Table 10.2 Training and generalisation performance of Learn++

Database	Class (1)	Class (−1)	Test performance (%)
S_1	132	68	72
S_2	125	75	82
S_3	163	37	85
S_4	104	96	86
Validate	143	57	–

variable to indicate the movement direction of the Dow Jones Index at time t. If the Dow Jones Index at time t is larger than that at time $t-1$, *Direction*is 1, otherwise, *Direction*is -1.

The model estimation selection process is then followed by an empirical evaluation which is based on the out-of-sample data. At this step, the comparative performance of the model is measured by the classification accuracy of the final hypothesis chosen for all given databases. The confidence of the algorithm on its own decision is used to evaluate the accuracy of the predicted closing value category. The first experiment implements a one step forward prediction of the next day's stock closing value. After predicting the next day's closing value this value is fed into a classification model to indicate the direction of movement for the stock prices. As discussed above, the database consisted of 1,476 instances of the Dow Jones average closing value during the period from January 2000 to November 2005; 1,000 instances are used for training and all the remaining instances are used for validation (Lunga and Marwala 2006a). The two binary classes are 1, indicating an upward direction of returns in Dow Jones stock, and -1 to indicate a predicted fall/downward direction of movement for the Dow Jones stock.

Four datasets S_1, S_2, S_3 and S_4, where each dataset included exactly one quarter of the entire training data, were provided to Learn++ in four training sessions for incremental learning. For each training session k ($k = 1, 2, 3, 4$) three weak hypothesis were produced by Learn++. Each hypothesis h_1, h_2, and h_3 of the kth training session was produced using a training subset TR_t and a testing subset TE_t. The weak-learner was a single hidden layer multi-layer perceptron with 15 hidden layer nodes and 1 output node with an MSE goal of 0.1. The testing set of data consisted of 476 instances that were used for validation purposes. On average, the multi-layer perceptron hypothesis, weak-learner, performed little over 50 %, which improved to over 80 % when the hypotheses were combined by making use of weighted majority voting. This improvement demonstrated the performance improvement property of Learn++, as inherited from Adaptive Boosting, on a given database. The data distribution and the percentage classification performance are given in Table 10.2 (Lunga and Marwala 2006b). The performances listed are on the validation data.

Table 10.3 gives an actual breakdown of correctly classified and misclassified instances falling into each confidence range after each training session. The trends of the confidence estimates after subsequent training sessions are given in Table 10.3. The desired outcome on the actual confidences is high to very high

Table 10.3 Confidence results

		VH	H	M	VL	L
Correctly classified	S_1	96	96	13	15	6
	S_2	104	104	22	17	14
	S_3	111	111	6	3	39
	S_4	101	101	42	12	4
Incorrectly classified	S_1	23	7	13	3	8
	S_2	27	0	1	3	4
	S_3	21	1	2	4	2
	S_4	24	0	2	2	0

Table 10.4 Confidence trends for Dow Jones

	Increasing steady	Decreasing
Correctly classified	119	8
Misclassified	16	24

confidences on correctly classified instances, and low to very low confidences on misclassified instances. The desired outcome on confidence trends is increasing or steady confidences on correctly classified instances, and decreasing confidences on misclassified instances, as new data is introduced.

The performance shown in Table 10.2 indicates that the algorithm is improving its generalization capacity as new data become available. The improvement is modest, however, as majority of the new information is already learned in the first training session. Table 10.4 indicates that the vast majority of correctly classified instances tend to have very high confidences, with continually improved confidences at consecutive training sessions (Lunga and Marwala 2006a).

While a considerable portion of misclassified instances also had high confidence for this database, the general desired trends of increased confidence on correctly classified instances and decreasing confidence on misclassified ones were notable and dominant, as shown in Table 10.3 (Lunga and Marwala 2006a).

10.6 Conclusions

In this chapter, an incremental learning procedure, Learn++, was applied to predict the financial markets movement direction. Learn++ is found to provide good results on adapting the weak-learner (MLP) into a strong learning algorithm that has confidence in all its decisions. The Learn++ procedure was found to evaluate the confidence of its own decisions. Generally, the majority of correctly classified cases had very high confidence approximations while lower confidence values were related with misclassified cases. Consequently, classification cases with low confidences can be further evaluated. In addition, the procedure also demonstrated increasing confidences in correctly classified instances and decreasing confidences in misclassified instances after successive training sessions.

References

Almaksour A, Anquetil E (2009) Fast incremental learning strategy driven by confusion reject for on-line handwriting recognition. In: Proceedings of the international conference on document analysis and recognition, Barcelona, 2009, pp 81–85

Austin PC, Lee DS, Steyerberg EW, Tu JV (2012) Regression trees for predicting mortality in patients with cardiovascular disease: what improvement is achieved by using ensemble-based methods? Biom J 54:657–673

Baraldi P, Razavi-Far R, Zio E (2011) Classifier-ensemble incremental-learning procedure for nuclear transient identification at different operational conditions. Reliab Eng Syst Saf 96: 480–488

Bishop C (1995) Neural networks for pattern recognition. Oxford University Press, Oxford

Bouchachia A (2011) Incremental learning with multi-level adaptation. Neurocomputing 74:1785–1799

Breiman L (1996) Bagging predictors. Mach Learn 24:123–140

Carpenter G, Grossberg S, Marhuzon N, Reynolds J, Rosen D (1992) ARTMAP: a neural network architecture for incremental learning supervised learning of analog multi-dimensional maps. IEEE Trans Neural Netw 3:678–713

Chen Y, Wong ML (2011) Optimizing stacking ensemble by an ant colony optimization approach. In: Proceedings of the genetic and evolutionary computation conference, Dublin, 2011, pp 7–8

Cho Y, Seong J-K, Jeong Y, Shin SY (2012) Individual subject classification for Alzheimer's disease based on incremental learning using a spatial frequency representation of cortical thickness data. Neuroimage 59:2217–2230

Ebrahimpour R, Babakhani K, AbbaszadehArani SAA, Masoudnia S (2012) Epileptic seizure detection using a neural network ensemble method and wavelet transform. Neural Netw World 22:291–310

Efron B (1979) Bootstrap methods: another look at the jackknife. Ann Stat 7:1–26

Erdem Z, Polikar R, Gurgen F, Yumusak N (2005) Reducing the effect of out-voting problem in ensemble based incremental support vector machines. Lect Note Comput Sci 3697:607–612

Folly KA (2011) Performance evaluation of power system stabilizers based on population-based incremental learning (PBIL) algorithm. Int J Electr Power Energy Syst 33:1279–1287

Freund Y, Schapire R (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55:119–139

Fu L, Hsu HH, Principe JC (1996) Incremental backpropagation networks. IEEE Trans Neural Netw 7:757–761

Ghimire B, Rogan J, Galiano V, Panday P, Neeti N (2012) An evaluation of bagging, boosting, and random forests for land-cover classification in Cape Cod, Massachusetts, USA. GISci Remote Sens 49:623–643

Halawani SM, Ahmad A (2012) Ensemble methods for prediction of Parkinson disease. Lect Note Comput Sci 7435:516–521

Hannah LA, Dunson DB (2012) Ensemble methods for convex regression with applications to geometric programming based circuit design. In: Proceedings of the 29th international conference on machine learning, Edinburgh, 2012, pp 369–376

Hansen LK, Salamon P (1990) Neural network ensembles. IEEE Trans Pattern Anal Mach Intell 12:993–1001

Higgins CH, Goodman RM (1991) Incremental learning for rule based neural network. In: Proceedings of the international joint conference on neural networks, Seattle, 1991, pp 875–880

Huang D, Yi Z, Pu X (2009) A new incremental PCA algorithm with application to visual learning and recognition. Neural Process Lett 30:171–185

Hulley G, Marwala T (2007) Genetic algorithm based incremental learning for optimal weight and classifier selection. In: Proceedings of the AIP conference, Sydney, Australia, pp 258–267

Jordan MJ, Jacobs RA (1994) Hierarchical mixtures of experts and the EM algorithm. Neural Comput 6:181–214

Khreich W, Granger E, Miri A, Sabourin RA (2009) A comparison of techniques for on-line incremental learning of HMM parameters in anomaly detection. In: Proceedings of the IEEE symposium on computational intelligence for security and defense applications, Ottawa,2009, pp 1–8

Khreich W, Granger E, Miri A, Sabourin R (2012) A survey of techniques for incremental learning of HMM parameters. Inf Sci 197:105–130

Kuncheva LI, Bezdek JC, Duin R (2001) Decision templates for multiple classifier fusion: an experimental comparison. Pattern Recognit 34:299–314

La L, Guo Q, Yang D, Cao Q (2012) Multiclass boosting with adaptive group-based k-NN and its application in text categorization. Math Probl Eng 2012:1–24

Leung M, Daouk H, Chen A (2000) Forecasting stock indices: a comparison of classification and level estimation models. Int J Forecast 16:173–190

Li H, Shen C (2008) Boosting the minimum margin: LP boost vs. ada boost. In: Proceedings of the digital image computing: techniques and applications, Canberra, Australia, pp 533–539

Li H, Zhang T, Qiu R, Ma L (2012) Grammar-based semi-supervised incremental learning in automatic speech recognition and labeling. Energy Procedia 17:1843–1849

Lienemann K, Plötz T, Fink GA (2009) Stacking for ensembles of local experts in metabonomic applications. Lect Note Comput Sci 5519:498–508

Louzada F, Ara A (2012) Bagging k-dependence probabilistic networks: an alternative powerful fraud detection tool. Expert Syst Appl 39:11583–11592

Lu G-F, Zou J, Wang Y (2012) Incremental learning of complete linear discriminant analysis for face recognition. Knowl-Based Syst 31:19–27

Lunga D, Marwala T (2006a) Time series analysis using fractal theory and on-line ensemble classifiers. Lect Note Comput Sci 4304:312–321

Lunga D, Marwala T (2006b) On-line forecasting of stock market movement direction using the improved incremental algorithm. Lect Note Comput Sci 4234:440–449

Martínez-Rego D, Pérez-Sánchez B, Fontenla-Romero O, Alonso-Betanzos A (2011) A robust incremental learning method for non-stationary environments. Neurocomputing 74:1800–1808

Marwala T (2007) Computational intelligence for modelling complex systems. Research India Publications, New Delhi

Marwala T (2009) Computational intelligence for missing data imputation, estimation and management: knowledge optimization techniques. IGI Global Publications, New York

Marwala T (2012) Condition monitoring using computational intelligence methods. Springer, London

Marwala T, Lagazio M (2011) Militarized conflict modeling using computational intelligence techniques. Springer, London

McCloskey M, Cohen N (1989) Catastrophic interference connectionist networks: the sequential learning problem. Psychol Learn Motiv 24:109–164

McNelis PD (2005) Neural networks in finance: gaining the predictive edge in the market. Elsevier Academic Press, Oxford

Mohamed S, Rubin D, Marwala T (2006) Multi-class protein sequence classification using fuzzy ARTMAP. In: Proceedings of the IEEE international conference on systems, man, and cybernetics, Taipei, 2006, pp 1676–1681

Mohamed S, Rubin D, Marwala T (2007) Incremental learning for classification of protein sequences. In: Proceedings of the IEEE international joint conference on neural networks, Orlando, 2007, pp 19–24

Muhlbaier M, Topalis A, Polikar R (2004) Learn++.MT: a new approach to incremental learning. In: Proceedings of the 5th international workshop on multiple classifier systems, Cagliari, 2004, pp 52–61

Nelwamondo FV, Marwala T (2007) Handling missing data from heteroskedastic and nonstationary data. Lect Note Comput Sci 4491:1293–1302

Nock R, Piro P, Nielsen F, Bel Haj Ali W, Barlaud M (2012) Boosting k-NN for categorization of natural scenes. Int J Comput Vis 100:294–314

Okada S, Kobayashi Y, Ishibashi S, Nishida T (2009) Incremental learning of gestures for human-robot interaction. AI Soc 25:155–168

Perez M, Featherston J, Marwala T, Scott LE, Stevens DM (2010) A population-based incremental learning approach to microarray gene expression feature selection. In: Proceedings of the IEEE 26th convention of electrical and electronic engineers, Eilat, 2010, pp 10–14

Polikar R (2000) Algorithms for enhancing pattern separability, feature selection and incremental learning with applications to gas sensing electronic noise systems. Ph.D. thesis, Iowa State University, Ames

Polikar R (2006) Ensemble based systems in decision making. IEEE Circuit Syst Mag 6:21–45

Polikar R, Byorick J, Krause S, Marino A, Moreton M (2002) Learn++: a classifier independent incremental learning algorithm for supervised neural network. Proc Int Jt Conf Neural Netw 2:1742–1747

Polikar R, Udpa L, Udpa S, Honavar V (2004) An incremental learning algorithm with confidence estimation for automated identification of NDE signals. Trans Ultrason Ferroelectr Freq Control 51:990–1001

Rogova G (1994) Combining the results of several neural network classifiers. Neural Netw 7:777–781

Schapire RE (1990) The strength of weak learnability. Mach Learn 5:197–227

Schapire RE, Freund Y, Bartlett P, Lee WS (1998) Boosting the margin: a new explanation for the effectiveness of voting methods. Ann Stat 26:51–1686

Sulzmann J-N, Fürnkranz J (2011) Rule stacking: an approach for compressing an ensemble of rule sets into a single classifier. Lect Note Comput Sci 6926:323–334

Syarif I, Zaluska E, Prugel-Bennett A, Wills G (2012) Application of bagging, boosting and stacking to intrusion detection. Lect Note Comput Sci 7376:593–602

Tang J, Shi Y, Zhou L, Zhang W (2008) Analog circuit fault diagnosis using Ada boost and SVM. In: Proceedings of the international conference on communications, circuits and systems, Fujian, China, pp 1184–1187

Tangruamsub S, Takada K, Hasegawa O (2012) A fast on-line incremental learning method for object detection and pose classification using voting and combined appearance modelling. Signal Process Image Commun 27:75–82

Tong J, Hu BX, Yang J (2012) Data assimilation methods for estimating a heterogeneous conductivity field by assimilating transient solute transport data via ensemble Kalman filter. Hydrol Process. doi:10.1002/hyp.9523

Topalov AV, Oniz Y, Kayacan E, Kaynak O (2011) Neuro-fuzzy control of antilock braking system using sliding mode incremental learning algorithm. Neurocomputing 74:1883–1893

Tscherepanow M, Kortkamp M, Kammer M (2011) A hierarchical ART network for the stable incremental learning of topological structures and associations from noisy data. Neural Netw 24:906–916

Vasquez D, Fraichard T, Laugier C (2009) Growing hidden markov models: an incremental tool for learning and predicting human and vehicle motion. Int J Robot Res 28:1486–1506

Vilakazi B (2007) Machine condition monitoring using artificial intelligence: the incremental learning and multi-agent system approach. M.Sc. thesis, University of the Witwatersrand, Johannesburg

Vilakazi CB, Marwala T (2007a) Incremental learning and its application to bushing condition monitoring. Lect Note Comput Sci 4491:1237–1246

Vilakazi CB, Marwala T (2007b) On-line incremental learning for high voltage bushing condition monitoring. In: Proceedings of the international joint conference on neural networks, Orlando, 2007, pp 2521–2526

Vilakazi B, Marwala T, Mautla R, Moloto E (2006) On-line bushing condition monitoring using computational intelligence. WSEAS Trans Power Syst 1:280–287

Wolpert DH (1992) Stacked generalization. Neural Netw 5:241–259

Xia H, Wu P, Hoi SCH, Jin R (2012) Boosting multi-kernel locality-sensitive hashing for scalable image retrieval. In: Proceedings of the international ACM SIGIR conference on research and development in information retrieval, Portland, Oregon, pp 55–64

Yamaguchi K, Yamaguchi N, Ishii N (1999) Incremental learning method with retrieving of interfered patterns. IEEE Trans Neural Netw 10:1351–1365

Yang S, Wang M, Jiao L (2011) Extreme and incremental learning based single-hidden-layer regularization ridgelet network. Neurocomputing 74:1809–1814

Zhang Y-X, Wang X-D, Yao X, Bi K (2012) HRRP recognition for polarization radar based on bagging-SVM dynamic ensemble. Syst Eng Electron 34:1366–1371