Artif Intell Rev (2018) 49:281–299
https://doi.org/10.1007/s10462-016-9526-2

A review of adaptive online learning for artificial neural networks

Beatriz Pérez-Sánchez[1] · **Oscar Fontenla-Romero**[1] · **Bertha Guijarro-Berdiñas**[1]

Published online: 22 October 2016

Abstract In real applications learning algorithms have to address several issues such as, huge amount of data, samples which arrive continuously and underlying data generation processes that evolve over time. Classical learning is not always appropriate to work in these environments since independent and indentically distributed data are assumed. Taking into account the requirements of the learning process, systems should be able to modify both their structures and their parameters. In this survey, our aim is to review the developed methodologies for adaptive learning with artificial neural networks, analyzing the strategies that have been traditionally applied over the years. We focus on sequential learning, the handling of the concept drift problem and the determination of the network structure. Despite the research in this field, there are currently no standard methods to deal with these environments and diverse issues remain an open problem.

Keywords Artificial neural networks · Online learning · Concept drift · Adaptive topology

1 Introduction

In real world problems, machine learning algorithms act in dynamic and evolving environments where the training data is flowing continuously or in separate blocks, such as financial analysis, meteorological data, fraud protection of bank cards, traffic monitoring, predictive customer behavior. This means that the information involved in the learning process is not completely available from the beginning but, rather, is continuously received and must be

✉ Beatriz Pérez-Sánchez
bperezs@udc.es

Oscar Fontenla-Romero
ofontenla@udc.es

Bertha Guijarro-Berdiñas
cibertha@udc.es

[1] Department of Computer Science, Faculty of Informatics, University of A Coruña, Campus de Elviña, 15071 A Coruña, Spain

processed sequentially in real time. This new information may affect the previously learned model and therefore, learning algorithms must be able to adapt dynamically as new data arrives. Moreover, classical batch learning presupposes identically distributed training data and a static nature of the world is assumed. Therefore, these kinds of learning algorithms learn the concept using available data and have to re-train when there are new samples (Esposito et al. 2004). This approach has several problems, the most important being its high demand of computational resources both spatial and temporal, an important handicap when dealing with high dimension data sets. Directly or indirectly, batch learning paradigm assumes the following constraints:

– Each time, the learning process needs to handle the whole training data set.
– There are no temporal restrictions to completely adjust the model.
– The input data do not suffer from changes, hence the learned model does not need further updates.

With all these limitations the applicability of machine learning is reduced significantly and online learning is becoming a good alternative to face the requirements of recent learning systems. Modern information systems has meant an increase in online learning (Minku et al. 2010). It is worth mentioning that the online learning concept can be found in literature referred to as: (1) the process underlying the data generation changes, (2) a huge amount of data is available, (3) data flows continuously, hence there is no consensus among researchers. The online models also known as incremental or sequential, obtain a model which is at least as precise as any other one trained with all examples. Moreover in dynamic environments, it is possible that data distribution varies along time leading to the well-known concept drift (Klinkenberg 2004; Widmer and Kubat 1996). Under some conditions, if the nonlinear dynamical system to be modeled presents periodic or recurrent trajectories where the concepts shift only among finitely many possibilities, the deterministic learning (DL) theory (Wang and Hill 2006) establishes that an accurate approximation of the system dynamics can be achieved by training locally neural networks, each one approximating a local region of the periodic trajectory. Therefore, along time, these learned neural networks can be recalled and reused for same or similar tasks without the need to retrain (Zeng et al. 2014, 2016; Zeng and Wang 2015). Nevertheless, in general concept drift is related to the possible variations in the output distribution even when the input remains stable. The model obtained through learning process should show two important characteristics despite their requirements being in conflict (Grossberg 1987; Robins 2004). On one hand, stability in order to keep significant knowledge and on other hand, plasticity to update the model when new relevant information is available. The ideal situation is that the system considers that new samples are more important than the older ones to model the current target concept (Kubat et al. 2004). For these reasons, it can be established that adaptation is a fundamental characteristic and learning systems should include some kind of adaptive mechanisms which allow them to act and to react in order to handle dynamic environments (Bouchachia et al. 2007). The research community has published important work to address learning in dynamic environments, thus it has become a widely studied field (Bottou 2004; LeCunn et al. 1998; Moller 1993; Rosenblatt 1958).

Since the nineties several neural networks architectures can be found in the literature, with feedforward neural networks (FNN) becoming the most widely applied structure thanks to their features such as adaptable architecture and representational potential. Despite these important advantages, several aspects remain unsolved. In dynamic environments one of the main key characteristics is the ability of the network to self-adapt, modifying both its control parameters and its structure according to the needs of the learning process. Consequently, different approaches can be found in the literature, some of them focused on parameter

adaptation (adjustment of weights and/or other parameters without changing the topology) whereas others aim for structural adaptation (adding or removing units and connections).

In this paper we present a review of the main strategies for adaptive neural networks, both parameter and structure aspects, which have been applied to treat dynamic environments. It is worth mentioning that our goal is not to provide a comprehensive review of articles but to establish the main strategies followed through the years to address the concept change and adaptation of the structure. This contribution is structured as follows. In Sect. 2 a brief introduction to the artificial neural networks (ANN) is presented emphasizing the assumptions of classical learning. Section 3 discusses some of the developed approaches to learn in an online mode. In Sect. 4 we review the main strategies for parameter adaptation in order to learn in presence of concept drift. Section 5 addresses the principal methods for network size determination. Section 6 points to the main challenges in this research field. Finally, in Sect. 7 some conclusions are given.

2 Artificial neural networks

Artificial neural networks (ANN) were introduced as alternative computing structures, created to replicate the functions of the human brain. Taking into account the description formulated by Haykin (1999), a neural network can be defined as a set of simple processing units whose union makes a parallel distributed processor that stores experiential knowledge and facilitates its use. These structures present very interesting properties and capabilities, as for example, an important computing power which allow us to solve complex problems in different application areas. In the recent literature, we have found different recommendations about neural networks models to solve tasks such as, regression problems, pattern classification or function approximation. Among them all, FNN are the most widely accepted thanks to their characteristics, as representational abilities or structure adaptability. Both multilayer perceptron (MLP) and radial basis function (RBF) networks are two examples of well-known feedforward neural networks. The backpropagation learning algorithm proposed by Rumelhart et al. (1986) is perhaps the most popular to train FNN and many variants were presented over the years. More advanced methods such as Levenberg–Marquardt (Hagan and Menhaj 1994; Levenberg 1944; Marquardt 1963), quasi-Newton (Bishop 1995) or conjugate gradient algorithms (Beale 1972; Moller 1993) are also very popular. These classical learning methods, based on gradient descent, assume the following ideas:

- As the batch learning paradigm is followed, whole training data set is managed during training and all weights are adjusted employing an accurate estimation of the error gradient vector. The modification of the network is made gradually by modifying the weights according to the direction of the gradient descent with respect to the error function.
- These methods search for appropriate weights in a fixed topology previously established. Therefore, this approach is useful only if the user selects an appropriate network for the learning problem on hand.

As a result, in order to be applicable in dynamic environments and face possible changes, these kinds of algorithms, which have been developed originally for static environments, have to be modified. As we previously commented and as shown in Fig. 1, learning algorithms must be modified to fulfill three main requirements: (1) capacity of working in an online mode, (2) ability to adjust their controlling parameters, and (3) capability to adapt their structures, all of them according to the requirements of the learning process.

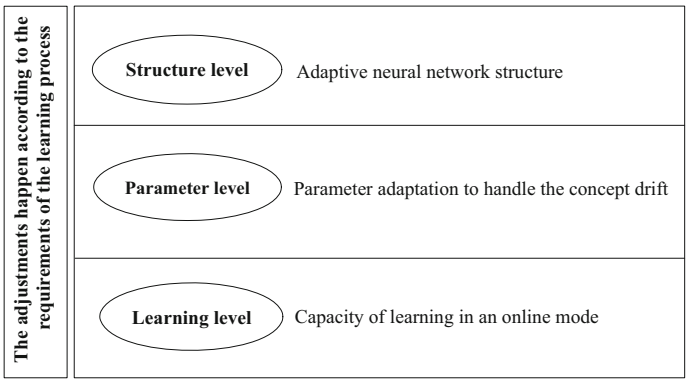

Fig. 1 Main requirements that should fulfill the neural networks learning algorithms to work in dynamic environments

3 Online learning algorithms

An online learning approach allows neural networks to solve dynamic real world problems since, in such contexts some kind of adaptation is indispensable. As we have previously mentioned, classical learning algorithms based on gradient descent operate in batch manner but they can be modified to adapt the weights of the neural network in a sequential manner, example by example. Online learning is preferable for large scale data sets and real time problems and it also avoids the problem of obtaining a local minimum. This issue has attracted a lot of attention in research field of neural networks since the early 1960s and several algorithms have been proposed.

Least-mean-square (LMS) (Widrow and Hoff 1960) employes a stochastic gradient-based method of steepest descent. LMS uses the estimates of gradient vector of the available data and includes an iterative mechanism to update the weights in the direction of the negative of the gradient vector. In case of highly correlated inputs, the recursive least-squares (RLS) method shows a faster convergence and a better behavior although the requirements of computational resources increase. A popular alternative is normalized least-mean-square (NLMS) (Nagumo and Noda 1967) which exhibits two advantages with respect to the original LMS. Firstly, a faster convergence for correlated and whitened input data and secondly a stable functioning for a range of values independently of the interrelations of input data (Goodwin and Sin 1984; Nagumo and Noda 1967). Other variants of the LMS based on kernel methods such as kernel least mean square (KLMS) (Liu et al. 2008a), kernel recursive least-squares (KRLS) (Engel et al. 2004) or more recently, the extended KRLS method (EX-KRLS) (Liu et al. 2009) have also been proposed.

Another approach is the online sequential extreme learning machine (OS-ELM), an online algorithm applicable to single hidden layer FNN (Huang et al. 2006; Liang and Huang 2006). This method proposed an unified framework scheme that allows the network to have different types of hidden units. OS-ELM consists of two phases, of the first being initialization where random values are assigned to input weights and then, a sequential learning phase to update the weights corresponding to the second layer. A more recent work described an online extreme learning machine combined with a time-varying neural network for learning non stationary data (Ye et al. 2013).

All the methods previously commented are only some examples of online learning algorithms for neural networks that have been proposed over the years. In Jain et al. (2014) a review of supervised neural networks with online learning capacities was presented. The authors provide a complete review about how to include the online learning paradigm and also show working approaches in real domains. Moreover, this review aims to inspire further research and development of more robust online learning networks for real problems.

4 Parameter adaptation and concept drift

In real world environments, data flows continuously and usually concept and data distribution change over time. A learning algorithm, with the ability of working in an online mode, could also include some mechanism to handle different scenarios which can be appear in a dynamic environment. This fact implies the need to automatically adjust its parameters. In those situations where data evolve, having some type of changes control is essential since an accurate decision model should be handled at each moment. An important problem is concept drift which is related to the possible variations in the output distribution even when the input remains stable (Gama et al. 2013). Different types of alterations can be found, for example, abrupt (moving suddenly from a context to another one) or gradual (through several intermediate steps). Moreover drift also appears associated to recurring or hidden contexts. Independently of the kind of drift, predictive models should include some mechanisms which allow the models to detect the alterations and deal with them, otherwise their potential will decrease. A suitable predictive model is able to:

1. *Adapt to concept drift as soon as possible*. The detection of the changes should be quick enough to adapt the behavior of the system and face new environments.
2. *Distinguish noise from changes*. An ideal learner should properly handle the noise without interpreting it as drift and self-adjust to the changes.
3. *Recognize and react to reoccurring contexts*. The use of previous experience is suitable to handle situations where old contexts and their corresponding concepts can reappear.

In recent machine learning literature, we can find three major trends to handle the different kinds of changes that can appear in the dynamics of the process (Ditzler et al. 2015; Gama 2010). Such categories, as shown in Fig. 2, are: (1) instance selection, (2) instance weighting and, (3) classifier ensembles. Below we present a brief synopsis of each of them.

(a) Instance selection is the most widely applied technique to handle concept drift. A moving window which encloses newer data is employed to learn the concept and adjust the

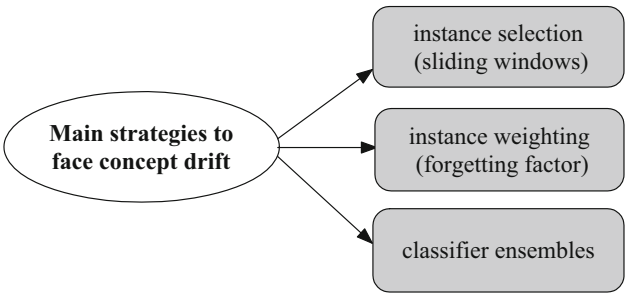

Fig. 2 Main strategies to face concept drift phenomenon

model. The length of this window has been considered as a critical aspect. Both static and changing options have been proposed however, both approaches present inconveniences. In Alippi and Roveri (2008) the window length is determined based on the expected change ratio but recent research affirms that determining the window size in an online adaptive mode is the best option (Alippi et al. 2011, 2012, 2013; Bifet and Gavalda 2006, 2007; Gama et al. 2004). The research developed by Kuncheva and Žliobaitė (2009) can be regarded as a step in the direction of determining an optimal window size in relation to the chosen classifier model and the properties of the streaming data.

(b) Instance weighting incorporates a factor that weighs the instances to obtain a balance between the information given by the newer examples and the old ones. Weighting criteria can be the age or relevancy of the instance with respect to its adequacy for the actual concept. Martínez-Rego et al. (2011) introduced a single-layer network for online learning which includes a forgetting factor to give monotonically increasing importance to new data. This model shows a suitable behavior in stationary contexts and it also has the ability of forgetting quickly and adapt to new contexts in changing environments. Plavidis et al. (2011) proposed an adaptive forgetting function based on the recursive least square adaptive filter for instance weighting. Ghazikhani et al. (2014) presented a forgetting function which is embedded into the neural network model to handle the non-stationary feature of the data.

(c) Classifier ensembles Classifier ensembles join several models in order to obtain a final solution. The ensemble can be extended including new models that store latest information corresponding to the current environment. Among the main advantages of this approach can be mentioned, for example that it tends to be more accurate than a single classifier due to the reduction in the variance of the error, or that it is more suitable to handle recurring environments due to the knowledge previously acquired being kept by the ensemble and hence, the ensemble can quickly adjust when a previous state appears again. The advantage of ensemble in nonstationary environments has been theoretically demonstrated, proving that an ensemble of classifiers provided more stable results than single classifiers (Ditzler et al. 2013, 2014). Moreover, recent research also indicate that the use of ensembles benefits the tracking of different rates of drift in data (Minku et al. 2010; Minku and Yao 2012). However, as a handicap it is worth mentioning that the fact of creating new models and storing the previous ones involves important computational and memory requirements. For these reasons the adaptation of the ensembles to changes can be too slow.

In Bouchachia (2011) the author recommended incremental learning with an adaptation scheme divided into three differentiated levels: the base classifiers self adjust in a natural way to possible changes in the environment, a contributive adaptation of the base learners which form the ensemble and finally, a structural adaptation of the ensemble. Other works in ensemble learning can be found in Brzezinski and Stephanowski (2014), Ditzler et al. (2013) and Elwell and Polikar (2011).

To sum up, learning algorithms should incorporate some change detection mechanism to deal with non-stationary environments. The most critical issue is to decide which technique should be employed. In Gama et al. (2013), the authors present a discussion on the best practice for performance assessment when learning is a continuous process emphasizing the use of predictive sequential error estimates by means of forgetting factors or sliding windows.

5 Structural adaptation

As was previously mentioned, the adaptation of the network implies changing not only the weights and biases, but eventually, also its architecture. The size of the neural network should fit the number and the complexity of the analyzed data. The unawareness about the appropriate size of the network to solve a problem presents several drawbacks:

- A basic model is not able to approximate correlated input data and overfitting could appear in case of a too complex model.
- The developer has to train several networks with different structures in order to know which is the smallest architecture with the ability to solve the problem at hand. As the decision is based on trial and error there is no guarantee that the selected number of hidden units is optimal and it is also computationally expensive.
- In cases where a large data set is available the number of hidden units required to face the problem increases so in these cases, reducing the requirements of computational resources is essential.

Among the reasons for finding optimal structure for a neural network we can mention among others, enhancing the predictions, improving the generalization capability and saving computational requirements especially when working with large data sets (Reitermanová 2008). The most critical issue, which has been the subject of many research papers, is how to choose an appropriate number of hidden layers and their respective quantity of units. Unlike traditional algorithms where the network structure has to be defined before the training process, in the adaptive approach the network architecture is constructed alongside the training process. The topology must be changed only if its capabilities are insufficient to satisfy the requirements of learning process. Up until now numerous studies focused on knowing the appropriate structure network and different strategies have been proposed. The general methods which have been applied are shown in Fig. 3 and briefly detailed as follows:

- *Constructive algorithms* start from a small network which later increases its size by adding hidden layers, nodes and connections until such time as neither performance improvements nor error requirements are obtained.
- *Pruning algorithms* employ a large network and along the training process irrelevant layers, units and connections are eliminated.
- *Hybrid methods* combine both previous approaches and they are considered as a promising alternative. The networks may be pruned after completion or interleaved with the constructive process.
- *Regularization techniques* add a penalty term to the error function to be minimized to encourage smoother network mappings and decrease the effect of network connections without importance. The difficulty is in choosing a suitable regularization parameter, usually a trial and error procedure is applied. A good approach can be to use the regularization framework together with constructive or pruning algorithms.

Generally it is considered that the pruning technique presents several drawbacks with respect to the constructive being the most important ones (Kwok and Yeung 1997):

1. In case of pruning algorithms it is impossible to know how large the initial network should be.
2. Constructive approach always searches for small network solutions first due to its incremental nature. This fact implies a greater computational saving than pruning approach, in which large efforts may be spent in deleting redundant units and weights.

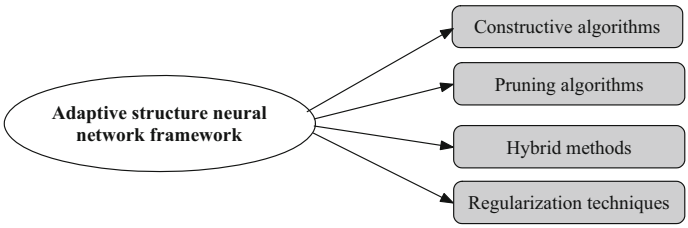

Fig. 3 General approach to self-adapt the neural network topology

3. As several suitable solutions can be achieved by means of different network structures it is easier that the constructive approach finds a smaller network than those obtained by pruning techniques.
4. When a unit or connection is eliminated, the error obtained by the pruned network is approximated by computational efficiency. This fact could introduce large errors.
5. Except for constructive algorithms, a suitable performance can only be obtained when the associated parameters have been properly tuned.

In following subsections, we review different procedures which follow the constructive, prune, hybrid and regularization approaches to obtain an appropriate network structure.

5.1 Constructive algorithms

Constructive algorithms handle a simple initial network (e.g., only one hidden layer that includes a single unit), which grows according to the needs of learning process. Algorithm 1 shows the basic procedure followed by the constructive approach to obtain a near optimal architecture.

Algorithm 1 Basic procedure followed by constructive approach

1. Starts from a minimal architecture. The number of input and output units are given by the problem. At the beginning only one unit is included in the hidden layer.
2. Initialize randomly all weights of the network within a small range.
3. Train the network on the training set by means of a specific learning algorithm.
4. Check the stop criterion. If it is satisfied, the training process is ended. Otherwise, continue.
5. Check the hidden neuron addition criterion. When it is fulfilled, continue. Otherwise, go to step 3.
6. Add one neuron to the hidden layer with random initial weights and go to step 3.

Three main issues have to be addressed to obtain an appropriate constructive algorithm:

1. *How to connect a new hidden unit to the existing network?* Taking into account the two possible options, enlarging an existing hidden layer or including a new one, the algorithm should decide which is the most appropriate one according to the individual case. Several studies confirm that, given enough nodes, single-hidden layer networks are universal approximators and can distinguish between arbitrarily complex decision regions in the input space (Hornik et al. 1989). However no theoretical bound is given for the number of required hidden units. Therefore other authors consider that constructive methods, which allow both width and depth development, are the best option taking into account the power of multi-hidden-layer networks as for example, faster learning (Chentouf and Jutten 1996).

2. *How to establish the weights of the new connections that result from the incorporation of a new unit?* In the constructive approach there are two general methods for training the network after adding a new hidden node: change all network's weights or only the weights of the new unit maintaining the remaining weights frozen.
3. *Which is the best criterion to decide when a network structure is suitable?* The more general method consists on either adding a new hidden node when the error does not reach the specified performance over a given period or testing if some performance criterion is reached, such as a minimum error.

Among the most common constructive algorithms we can mention adaptively constructing multilayer FNN (Ma and Khorasani 2003), enhanced incremental extreme learning machine (EI-ELM) (Huang and Chen 2008), repair radial basis function neural networks (RRBF) (Qiao and Han 2010) and extreme learning machine with adaptive growth of hidden nodes (AG-ELM) (Zhang et al. 2012).

Ma and Khorasani (2003) proposed a strategy to incrementally modify the neural network structure taking into account that such modifications will be made according to the requirements of the learning process. The idea is to adjust the architecture only when employing the current structure is not able to decrease the residual error.

García-Pedrajas and Ortiz-Boyer (2007) applied a cooperative evolutionary process which facilitates a modular construction of the network. Previously constructed modular networks can be combined with more recent ones in order to obtain better approaches. The methodology allows combining previously evolved networks with more recent modules to achieve a better approach.

Huang and Chen (2008) presented the enhanced incremental extreme learning machine (EI-ELM). Firstly, the method randomly generates several units and later the selection of the most appropriate one is based on the largest reduction of the residual error that can be achieved.

Another approach is the repair radial basis function algorithm (RRBF) (Qiao and Han 2010). The procedure begins with a randomly established prototype and it is carried out in two different stages which employ repair and adjustment strategies respectively. Firstly, the sensitivity analysis of the network's output is considered to determine when modifying the architecture. The structure is only repaired when the prototype does not satisfy the requirements. Secondly, an adjustment strategy is applied to improve the abilities of the model by revising all weights of the network.

Lastly, Zhang et al. (2012) proposed an extreme learning machine algorithm which includes a new strategy for automatic designing of the network (AG-ELM). The number of hidden units is adaptively determined and the current structure could be replaced by a newer and easier architecture which exhibits a better behavior.

More recent constructive approaches can be found in the literature, so Subirats et al. (2012) developed a novel constructive learning algorithm addressed by the thermal perceptron rule that guarantees the stability of the stored knowledge while the structure grows and the units compete for input information. After adding units to the network and thanks to the competition strategy, older neurons can still learn provided that new information is similar to their previously stored knowledge (Ortega-Zamorano et al. 2014, 2015).

A novel online learning algorithm for feedforward neural networks was proposed by Pérez-Sánchez et al. (2013, 2014, 2015). This method exhibits important characteristics among which we can mention for example (1) it is able to work both in static and dynamic environments, (2) the incorporation of new units does not alter the knowledge previously

acquired. Moreover, the method incorporates a mechanism to control the adaptation of the network topology which is based on the Vapnik–Chervonenkis dimension.

Other recent research can be found in Bertini Junior and Nicoletti (2016), Qiao et al. (2016), and Wang et al. (2015b). Constructive algorithms are considered efficient, however, it is worth noting that there are some issues that should be overcome. We can mention, among them, the need for establishing a criteria for stopping the addition of hidden units (see Sect. 5.5), the possibility of obtaining a suboptimal network due to the greedy approach employed by the majority of the algorithms, or the difficulties to obtain a suitable generalization power if the associated parameters are not correctly tuned.

5.2 Pruning algorithms

Pruning techniques start from a large network structure (actually its size is higher than necessary) which is reduced by means of removing unnecessary units and connections. When a large structure is employed, the influence of initial status and local minima are significantly reduced. Algorithm 2 shows the basic procedure followed by the pruning approach.

Algorithm 2 Basic procedure followed by pruning approach

1. Start from a large architecture. The number of units for input and output layers are given by the problem whereas the hidden layer contains a large number of units.
2. All weights of the network are randomly initialized within a short range.
3. Train the network until the minimum of an error function is reached or a pruning indicator triggers.
4. Prune one or several less significant hidden units.
5. Retrain the pruned network until its earlier error is achieved.
6. Repeat the steps 4 and 5 until the pruned network could not achieve its previous error level after retraining.

Many pruning techniques have been developed over the years, such as, optimal brain damage (OBD) (Cun et al. 1990), optimal brain surgeon (OBS) (Hassibi and Stork 1993), subset-based training and pruning (SBTP) (Xu and Ho 2006), extended Fourier amplitude sensitivity test method (EFAST) (Lauret et al. 2006) or optimally pruned extreme learning machine (OP-ELM) (Yoan et al. 2010). In what follows, a short introduction for each of them is given.

Cun et al. (1990) introduced the OBD technique for reducing the size of a network by selectively deleting weights. This strategy is based on the idea that starting from an initial network, it is possible to obtain a less complex model which exhibits a similar or better performance. In order to improve the the network pruning process (Hassibi and Stork 1993) considered including the information provided by the second order derivatives of the error function. The aim is to improve generalization and to increase the speed of training. The OBS algorithm is significantly better than optimal brain damage (OBD) which can often remove the wrong weights.

SBTP algorithm is based on the connection between node dependence and Jacobian rank deficiency and it was proposed by Xu and Ho (2006). At each learning iteration, dependent nodes are located by applying the orthogonal factorization with column permutation to the outputs of the hidden units of the same layer. Later, the output weights are set to zero for dependent units whereas in the case of independent neurons, they are updated by means of the Levenberg Marquardt algorithm. Finally, the unnecesary units are removed using OBS method.

Another approach is the EFAST (Lauret et al. 2006), a method which employes the Fourier decomposition of the variance of the model output to determine which is the significance of the hidden nodes. Each of these units has an associated ratio that facilitates its ranking. This measure is employed as criterion to select which units are unnecessary.

Finally, Yoan et al. (2010) started from the original extreme learning machine algorithm and developed the optimally pruned extreme learning machine (OP-ELM) methodology. This new approach includes a process for pruning units leading to a more robust overall method. The suitable number of hidden units is established by means of a leave-one-out criterion.

In addition, an approach able to recognize which are unnecessary hidden units and achieve an optimal structure is proposed in Augasta and Kathirvalavakumar (2011). For that, a new relevance measure, calculated by the sigmoidal activation value of a node and all weights of its output links, is introduced. When this significance value falls below a previously established threshold the associated units should be removed. Another more recent approach was presented in Thomas and Suhner (2015), the authors proposed a novel approach to determine the optimal structure based on variance sensitivity analysis, and prunes the different types of units (hidden neurons, inputs and weights) sequentially. The stop criterion is based on a performance evaluation of the network results from both the learning and validation datasets. Four variants which use two different estimators of the variance are proposed. In Augasta and Kathirvalavakumar (2013) a survey of existing pruning techniques for neural network's optimization is presented including a discussion about their advantages and drawbacks.

As was previously discussed, pruning techniques can remove unnecessary units and connections. In this regard, Bondarenko et al. (2015) compare nodes versus weights pruning algorithms. This experimental study shows that both types of pruning simplify network structure but the trimming of units is the preferred form. Finally, this research concludes that the approximation based on weights pruning is able to reach a good performance at cost of more complex network structure and higher computational time.

5.3 Hybrid algorithms

A hybrid method combines growing and pruning techniques and is considered a promising alternative. In this focus the networks are pruned after completion or interleaved with the constructive process. Algorithm 3 shows the basic procedure considering the trim after the constructive phase.

Algorithm 3 Basic procedure followed by a growing and pruning approach

1. Determine the number of hidden units using a constructive algorithm (see Algorithm 1).
2. Calculate a given relevance indicator of each hidden unit.
3. If it is necessary, prune the least significant hidden units.
4. Retrain the pruned network until achieving its previous error.
5. Repeat the steps 2 and 3 until the pruned network could not achieve its previous error level after retraining.

Fritzke (1994) introduced the growing cell structure (GCS), a self-organizing neural network that includes two different versions: one for unsupervised learning and another for supervised learning. The former controls the modifications of the structure including the possibility of removing units. Vector quantization, clustering or data visualization are some of its possible applications. The latter version employes a self-organizing neural network combined with a radial basis function approach and allows carrying out, at the same time,

the supervised training of the weights and the locating of the radial basis function units. The decision of where to add new units is based on the current classification error.

Huang et al. (2005) developed a generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation based on the significance concept. From a statistical point of view, the significance of a neuron is related to its associated information and its contribution over the global performance of the RBF network. This significance measure is considered both for growing and for pruning methods. In the case where the significance value of a specific unit is over an established learning accuracy, this neuron is added to the network. Otherwise, the unit will be removed.

Hsu (2008) developed the adaptive growing and pruning neural network (AGPNNC) system for a linear piezo-electric ceramic motor. Taking into account the goal of the system, imitating the behavior of an ideal computation controller, an online learning algorithm for a self constructing neural network is applied.

Narasimha et al. (2008) presented an adaptive algorithm which allows networks to be continually pruned during the growing process. In a first phase, a growing scheme allows adding new units and later, the network is pruned thanks to a method that employes orthogonal least squares. Finally, hidden units are ranked according to their suitability and the worst classified ones will be pruned. As handicaps, high computational requirements and the fact of not being strictly sequential, should be mentioned.

Huang and Du (2008) focused on radial basis probabilistic neural networks (RBPNN) and proposed a methodology for optimizing the network structure. The process is divided in two differenciate phases. Firstly, the selection of the initial hidden layer centers of the network is made by means of a minimum volume covering hyper spheres algorithm. Later, for getting a deeper optimization of the network structure a combination of the recursive orthogonal least square and the particle swarm optimization algorithms is applied.

Another improved hybrid algorithm is the adaptive merging and growing algorithm (AMGA) (Islam et al. 2009) which merges and adds units alongside the training process. AMGA adds hidden neurons by splitting existing hidden neurons and prunes hidden neurons by merging correlated hidden neurons. Thanks to this fact the number of retrainings needed after modifying the structure is reduced.

Among recent hybrid approaches found in the literature we can mention the following. Han and Qiao (2013) proposed a hybrid approach for single-layer feedforward neural network. Fourier decomposition of the networks's output variance is applied to calculate the contribution ratio of each hidden unit. This measure allows determining the suitable number of hidden units. The idea is to provide a full structure optimization methodology by means of a combination between the error reparation and sensitivity analysis.

de Jesus Rubio and Perez-Cruz (2014) addressed a stable evolving intelligent system for modelling nonlinear systems with dead-zone input. The method comes from recursive building of hidden units, parameter estimation and hidden neuron based model structure learning. Input and output pairs are jointed in clusters and for each them only one hidden unit is employed. When the distance between an instance and the cluster center falls below an established value, that instance is assigned to the closest cluster, otherwise becomes a new cluster center. Finally, after certain number of iterations, the least used units are pruned.

Marques Silva et al. (2014) introduced an evolving neural fuzzy modeling approach constructed upon a fuzzy neural network model based on the neo-fuzzy neuron which was introduced in Yamakawa et al. (1992). The procedure starts from choosing initial values of the membership degrees of each input, finds the most active function and adjusts its

value. The method controls when including or removing a new function in base of its activity.

Encouraged by multiple model approaches (Liu et al. 2008a) and localized modeling (Gregorcic and Lightbody 2007), Qiao et al. (2014) presented an online self-adaptive modular neural network, known as OSAMNN. The method starts from zero subnetworks and updates the centers of the radial basis functions thanks to a single-pass subtractive cluster approach. The complexity of the model is controlled by means of growing and merging procedures, adjusting the centers of the neurons according to the environment changes and maintaining a suitable model complexity.

5.4 Regularization techniques

Regularization techniques include a penalty term to the cost function with the aim of less important connections convergence to zero. It is worth mentioning that these methods are not able to automatically fix the structure of the network and a good approach is to use the regularization framework together with some of the previously mentioned techniques. A suitable selection of regularization parameter is an important handicap and usually it is determined by means of trial and error methods. In literature different selection techniques can be found in both deterministic and stochastic settings (Bauer and Lukas 2011). The effective of the weight decay method has been experimentally shown and the convergence for the gradient method with a weight decay term has been theoretically proved (Shao and Zheng 2011; Yu and Chen 2012; Zhang et al. 2009, 2012).

Recently, smoothing regularization methods are proposed for training feedforwards neural networks (Fan et al. 2014; Wu et al. 2014). Moreover, in Peng et al. (2015) a comparative among different regularization techniques is carried out whereas in Wang et al. (2015a) the authors review distinct performance aspects associated to the use of different penalization terms.

5.5 Criteria to stop adding or pruning hidden units

Independently of the strategy followed to construct the network it is necessary to decide when the structure is appropriate for solving a given problem. In the literature we can find several theoretical limits which come from different methods and approximations in order to control the number of hidden units. Among others, we can mention the following:

- Criteria based on the training error, when it is less than a threshold or flats out. However, it is well known that the training error is biased. Alternatively, it could depend on a separate test set, or on more complicated cross-validation or boot-strapping methods.
- Singular value decomposition (SVD) approach estimates the significance associated to an increment of a hidden unit along constructive/destructive process (Teoh et al. 2006). The approach considers that a determinate amount of units is sufficient when the contribution of each new unit falls bellow a predeterminate threshold. Signal to noise ratio (SNR) was also used to discover the optimal number of neurons, Liu et al. (2007, 2008b) assume that the training data may come with white gaussian noise at an unknown level. In order to have a clear indication of overfitting, the error signal is calculated and its levels of signal and noise can be obtained. The ratio of the signal energy level over the noise energy level is defined as the signal to noise ratio. The authors demonstrated that on one hand, the SNR quantitatively identifies overfitting and on the other hand, the obtained optimum point shows a divergence between the train and validation errors, at the same time the validation error reaching its minimum value.

– Other approaches founded on geometrical techniques and information entropy have been applied. For example, Baum and Haussler (1989) obtained different limits which come from the number of training instances employing networks composed of linear threshold networks. The Akaike information criterion (AIC), Bayesian information criterion (BIC), root mean squared error (RMSE) and Mean Absolute Percentage Error (MAPE) have also been employed in model selection. However, there are comparative studies that analyze these basis criteria and their variations to conclude that there is no best method (Qi and Zhang 2001). Association of these different criteria have also been presented (Egrioglu et al. 2008). Camargo and Yoneyama (2001) introduced a strategy for estimating the suitable number of units by means of Chebyshev polinomials and earlier works from Scarselli and Tsoi (1998). A fraction of these bounds proposed in previous research come from the Vapnik and Chervonenkis theory (Vapnik 1998).

Despite the numerous rules or criteria, which can be found in the literature to determine when to stop the building of the network, it is worth mentioning that they have been conceived for learning in batch mode. From the perspective of online learning, some of the previous rules are not applicable and others have to be adapted. To date the available criteria are not enough and the authors are not aware of an efficient method for determining the optimal network architecture for a problem at hand. Therefore, nowadays *which is the optimal neural network structure?* continues to be an unanswered question (Sharma and Chandra 2010).

6 Challenges and future research

Over the last decades neural networks has been widely employed for appropriately facing many real world applications. Learning in dynamic environments is a growing research field in machine learning due to the current needs of real applications that have to manage big volumes of data and process data in real time. As a consequence, effective methods with capacity to track the evolution of the data and self-adapt to changes are required. In spite of all the research made in this field there are still problems that involve important restrictions and need in-depth study. Among them, the following can be highlighted:

– Current methods are not able to identify and explain appropriately when, how, and where concept drift occurs.
– Most of the techniques are based on numerical data distributions meaning they are not able to tackle data with uncertainty associated.
– The concurrent detection of concept drift and noise becomes cumbersome

Furthermore, nowadays new challenges and trends arise:

– A particularly challenging problem is related to those scenarios where labeled samples are only available at the beginning of the process followed by unlabeled data drawn from a different distribution. Research on this topic is still at an early stage.
– Class imbalance problem has been thoroughly studied for stationary scenarios as opposed to the non stationary ones (He and Garcia 2009). Recent works focus on drifts of the minority class and specialized evaluation methods (Wang et al. 2015b). The imbalance problem has also encouraged the study other types of changes (Gama et al. 2014).
– Another problem is dealing with verification latency that occurs if the labels do not become available immediately as the next batch of data arrives. A mechanism to propagate class information forward through several time steps of unlabeled data are required. In Ditzler et al. (2015) a survey of attempts to face this problem can be found.

– Other studies are related to more complex representations of the samples such as, ordinal classification or multi-labeled outputs.
– Most of the developed methods have not been evaluated on large scale real problems

Learning for neural networks in non stationary environments still present important open problems (Ditzler et al. 2015; Krempl et al. 2014) and their research continue to be very active today. This is evident by the topics broached in some of the most important conferences like the *International Joint Conference on Artificial Neural Networks* (IJCNN), the *International Conference on Machine Learning* (ICML), the *European Conference on Machine Learning* (ECML) or the *Conference on Neural Information Processing Systems* (NIPS) in which during the last editions several tutorials, workshops and sessions have been organized around concept drift and domain adaptation; learning in dynamic environments or incremental machine learning; learning from data streams in large-scale evolving environments and challenges, methods and applications.

7 Conclusion

In this paper, we focus on the topic of learning in dynamic environments. In many real applications, learning algorithms have to work with huge amount of data, samples which arrive continuously or underlying data generation processes that evolve over time. Neural network models present very interesting abilities such as, an important computing power which allow us to solve complex problems in different application areas. However, classical learning algorithms for neural networks are not appropriate to deal with dynamic environments and an online focus is required. Moreover, the adaptation of the networks depends on the learning process needs. This adaptation implies changing not only the weights and biases but also their topologies. In order to develop learning algorithms for neural networks able to deal with dynamic environments there are several issues which deserve particular attention.

On one hand, we focused on possible trend changes result of variations in the distribution of the input data. We presented a brief review about how to face the learning when the process is affected by drift, introducing the most applied techniques to handle these types of environments. Another important challenge is to use a network with an adequate structure for solving the problem on hand. Unlike the traditional algorithms where the network structure has to be defined before the training process, in the adaptive approach the network architecture is constructed alongside the training process. We reviewed the general strategies which have been applied for constructing an optimal structure, giving their basic procedures, and analyzing their advantages, drawbacks and offering a summary of the researches developed.

In spite of the extensive research in this field, nowadays there are no standard methods to handle these types of environments and different issues, i.e., which is the most suitable number of hidden neurons, remains an open problem. Therefore, this topic definitely deserves more attention in the future.

Acknowledgements The authors would like to thank support for this work from the Xunta de Galicia (Grant code GRC2014/035) and the Secretaría de Estado de Investigación of the Spanish Government (Grant code TIN2015-65069), all partially supported by the European Union ERDF funds.

References

Alippi C, Roveri M (2008) Just-in-time adaptive classifiers—part II: designing the classifier. IEEE Trans Neural Netw 19(12):2053–2064

Alippi C, Boracchi G, Roveri M (2011) A just-in-time adaptive classification systems based on the intersection of confidence intervals rule. Neural Netw 24(8):791–800

Alippi C, Boracchi G, Roveri M (2012) Just-in-time ensemble of classifiers. In: Proceedings of international joint conference on neural networks (IJCNN'12), pp 1–8

Alippi C, Boracchi G, Roveri M (2013) Just-in-time classifiers for recurrent concepts. IEEE Trans Neural Netw Learn Syst 24(4):620–634

Augasta MG, Kathirvalavakumar T (2011) A novel pruning algorithm for optimizing feedforward neural network of classification problems. Neural Process Lett 34:241–258

Augasta MG, Kathirvalavakumar T (2013) Pruning algorithms of neural networks a comparative study. Cent Eur J Comp Sci 3(3):105–115

Bauer F, Lukas MA (2011) Comparing parameter choice methods for regularization of ill-posed problems. Math Comput Simul 81:1795–1841

Baum EB, Haussler D (1989) What size net gives valid generalization? Neural Comput 1:151–160

Beale EM (1972) A derivation of conjugate gradients, numerical methods for nonlinear optimization. Academic Press, New York

Bertini Junior JR, Nicoletti MC (2016) Enhancing constructive neural networks performance using functionally expanded input data. J Artif Intell Soft Comput Res 6(2):119–131

Bifet A, Gavalda R (2006) Kalman filters and adaptive windows for learning in data streams. In: Proceedings of international conference discovery science, pp 29–40

Bifet A, Gavalda R (2007) Learning from time-changing data with adaptive windowing. In: Proceddings of SIAM international conference on data mining (SDM 2007)

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

Bondarenko A, Borisov A, Aleksejeva L (2015) Neurons vs weights pruning in artificial neural networks. In: Proceedings of the 10th international scientific and practical conference, vol III, pp 22–28

Bottou L (2004) Stochastic learning. Adv Lect Mach Learn Lect Notes Artif Intell 3176:146–168

Bouchachia A (2011) Incremental learning with multi-level adaptation. Neurocomputing 74(11):1785–1799

Bouchachia A, Gabrys B, Sahel Z (2007) Overview of some incremental learning algorithms. In: Proceedings of the IEEE international conference on fuzzy systems, pp 1–6

Brzezinski D, Stephanowski J (2014) Reacting to different types of concept drift: the accuracy updated ensemble algorithm. IEEE Trans Neural Netw Learn Syst 25(1):81–94

Camargo LS, Yoneyama T (2001) Specification of training sets and the number of hidden neurons for multilayer perceptrons. Neural Comput 13(12):2673–2680

Chentouf R, Jutten C (1996) DWINA: depth and width incremental neural algorithm. In: Proceedings of the IEEE international conference on neural networks, pp 153–158

Cun YL, Denker JS, Solla SA (1990) Optimal brain damage. Adv Neural Inf Process 2:598–605

de Jesus Rubio J, Perez-Cruz H (2014) Evolving intelligent system for the modelling of nonlinear systems with dead-zone input. Appl Soft Comput 14(Part B):289–304

Ditzler G, Rosen G, Polikar R (2013) Discounted expert weighting for concept drift. In: IEEE symposium on computational intelligence in dynamic and uncertain environments (CIDUE'13), pp 61–67

Ditzler G, Rosen G, Polikar R (2014) Domain adaptation bounds for multiple expert systems under concept drift. In: International joint conference on neural networks (IJCNN'14), pp 595–601

Ditzler G, Roveri M, Alippi C, Polikar R (2015) Learning in nonstationary environments: a survey. IEEE Comput Intell Mag 10(4):12–25

Egrioglu E, Aladag CH, Gunay S (2008) A new model selection strategy in artificial neural networks. Appl Math Comput 195:591–597

Elwell R, Polikar R (2011) Incremental learning of concept drift in nonstationary environments. IEEE Trans Neural Netw 22(10):1517–1531

Engel Y, Mannor S, Meir R (2004) The kernel recursive least-squares algorithm. IEEE Trans Signal Process 52(8):2275–2285

Esposito F, Ferilli S, Fanizzi N, Basile T, Mauro MD (2004) Incremental learning and concept drift in INTHELEX. Intell Data Anal 8(3):213–237

Fan Q, Zurada JM, Wu W (2014) Convergence of online gradient method for feedforward neural networks with smoothing $l_{1/2}$ regularization penalty. Neural Netw 50:72–78

Fritzke B (1994) Growing cell structures a self-organizing network for unsupervised and supervised learning. Neural Netw 7(9):1441–1460

Gama J (2010) Knowledge discovery from data streams. Chapman and Hall/CRC, Boca Raton

Gama J, Medas P, Castillo G, Rodrigues P (2004) Learning with drift detection. In: Proceedings in adavances artificial intelligence (SBIA 2004), pp 586–295
Gama J, Sebastiao R, Pereira Rodrigues P (2013) On evaluating stream learning algorithms. Mach Learn 90(3):317–346
Gama J, Žliobaitė I, Bifet A, Pechenizkiy M, Bouchachia A (2014) A survey on concept drift adaptation. ACM Comput Surv 46(4):44:1–44:37
García-Pedrajas N, Ortiz-Boyer D (2007) A cooperative constructive method for neural networks for pattern recognition. Pattern Recognit 40(1):80–98
Ghazikhani A, Monsefi R, Sadoghi Yazdi H (2014) Online neural network model for non-stationary and imbalanced data stream classification. Int J Mach Learn Cybernet 5(1):51–62
Goodwin GC, Sin KS (1984) Adaptive filtering, prediction and control. Prentice-Hall, Englewood Cliffs
Gregorcic G, Lightbody G (2007) Local model network identification with gaussian processes. IEEE Trans Neural Netw 18:1404–1423
Grossberg S (1987) Competitive learning: from interactive activation to adaptive resonance. Cogn Sci 11(1):23–63
Hagan MT, Menhaj M (1994) Training feedforward networks with the marquardt algorithm. IEEE Trans Neural Netw 5(6):989–993
Han H-G, Qiao J-F (2013) A structure optimisation algorithm for feedforward neural network construction. Neurocomputing 99:347–357
Hassibi B, Stork DG (1993) Second-order derivatives for network pruning: optimal brain surgeon. Adv Neural Inf Process Syst 5:164–171
Haykin S (1999) Neural networks: a comprehensive foundation. Prentice Hall, New Jersey
He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Knowl Data Eng 21(9):1263–1284
Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Netw 2(5):359–366
Hsu CF (2008) Adaptive growing-and-pruning neural network control for a linear piezoelectric ceramic motor. Eng Appl Artif Intell 21(8):1153–1163
Huang G-B, Chen L (2008) Enhanced random search based incremental extreme learning machine. Neurocomputing 71(16–18):3460–3468
Huang DS, Du JX (2008) A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks. IEEE Trans Neural Netw 19(12):2099–2115
Huang GB, Saratchandran P, Sundararajan N (2005) Generalised growing and pruning RBF (GGAP-RBF) neural network for function approximation. IEEE Trans Neural Netw 16(1):57–67
Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theorey and applications. NeuroComputing 70:489–501
Islam MM, Sattar MA, Amin MF, Yao X, Murase K (2009) A new adaptive merging and growing algorithm for designing artificial neural networks. IEEE Trans Syst Man Cybern 39(3):705–722
Jain LC, Seera M, Lim CP, Balasubramaniam P (2014) A review of online learning in supervised neural networks. Neural Comput Appl 25:491–509
Klinkenberg R (2004) Learning drifting concepts: example selection vs. example weighting. Intell Data Anal 8(3):281–300
Krempl G, Žliobaitė I, Brzeziński D, Hüllermeier E, Last M, Lemaire V, Noack T, Shaker A, Sievi S, Spiliopoulou M, Stefanowski J (2014) Open challenges for data stream mining research. SIGKDD Explor 16(1):1–10
Kubat M, Gamma J, Utgoff P (2004) Incremental learning and concept drift, editor's introduction: guest-editorial. Intell Data Anal 8(3):211–212
Kuncheva L, Žliobaitė I (2009) On the window size for classification in changing environments. Intell Data Anal 13(6):861–872
Kwok T-Y, Yeung D-Y (1997) Constructive algorihtms for structure learning in feedforward neural networks for regresion problems. IEEE Trans Neural Netw 8(3):630–645
Lauret P, Fock E, Mara TA (2006) A node pruning algorithm based on a fourier amplitude sensitivity test method. IEEE Trans Neural Netw 17(2):273–293
LeCunn Y, Bottou L, Orr G, Müller K-R (1998) Efficient backprop. Neural Netw Tricks Trade 1524:9–50
Levenberg K (1944) A method for the solution of certain non-linear problems in least squares. Q J Appl Math 2(2):164–168
Liang N-Y, Huang G-B (2006) A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans Neural Netw 17(6):1411–1423
Liu Y, Starzyk A, Zhu Z (2007) Optimizing number of hidden neurons in neural networks. In: Proceedings of the artificial intelligence and applications (AIAP'07), pp 121–126

Liu W, Pokharel PP, Principe JC (2008a) The kernel least-mean-square algorithm. IEEE Trans Signal Process 56(2):543–554
Liu Y, Starzyk A, Zhu Z (2008b) Optimized approximation algorithm in neural networks without overfitting. IEEE Trans Neural Netw 19(6):983–995
Liu W, Park I, Principe JC (2009) Extended kernel recursive least squares algorithm. IEEE Trans Signal Process 57(10):3801–3814
Ma L, Khorasani K (2003) A new strategy for adaptively constructing multilayer feedforward neural networks. Neurocomputing 51:361–385
Marquardt DW (1963) An algorithm for least-squares estimation of non-linear parameters. J Soc Ind Appl Math 11(2):431–441
Marques Silva A, Caminhasa W, Lemosa A, Gomide F (2014) A fast learning algorithm for evolving neo-fuzzy neuron. Appl Soft Comput 14(B):194–209
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
Minku LL, White AP, Yao X (2010) The impact of diversity on on-line ensemble learning in the presence of concept drift. IEEE Trans Knowl Data Eng 22:730–742
Minku L, Yao X (2012) Ddd: a new ensemble approach for dealing with concept drift. IEEE Trans Knowl Data Eng 24(4):619–633
Moller M (1993) Supervised learning on large redundant training sets. Int J Neural Syst 4(1):15–25
Nagumo J, Noda A (1967) A learning method for system identification. IEEE Trans Autom Control 12:283–287
Narasimha PL, Delashmit WH, Manry MT, Li J, Maldonado F (2008) An integrated growing-pruning method for feedforward network training. Neurocomputing 71(13–15):2831–2847
Ortega-Zamorano F, Jerez J, Urda D, Luque-Baena R, Franco L (2014) Fpga implementation of the C-MANTEC neural networks constructive algorithm. IEEE Trans Ind Inf 10(2):1154–1161
Ortega-Zamorano F, Jerez J, Jurez G, Franco L (2015) Fpga implmentation comparison between c-mantec and back propagation. In: International workshop on artificial neural networks (IWANN 2015), vol Part II of LNCS, pp 197–208
Peng H, Mou L, Li G, Chen Y, Lu Y, Jin Z (2015) A comparative study on regularization strategies for embedding-based neural networks. In: Proceedings of the conference on empirical methods in natural language processing (EMNLP 2015), pp 2106–2111
Pérez-Sánchez B, Fontenla-Romero O, Guijarro-Berdiñas B, Martínez-Rego D (2013) An online learning algorithm for adaptable topologies of neural networks. Expert Syst Appl 40:7294–7304
Pérez-Sánchez B, Fontenla-Romero O, Guijarro-Berdiñas B (2014) Self-adaptive topology neural network for online incremental learning. In: Proceedings of the international conference on agents and artificial intelligence (ICAART'14), pp 94–101
Pérez-Sánchez B, Fontenla-Romero O, Guijarro-Berdiñas B (2015) Adaptive neural topology based on Vapnik–Chervonenkis dimension. In: Lecture Notes in Artificial Intelligence (in press)
Plavidis NG, Tasoulis DK, Adams NM, Hand DJ (2011) Landa perceptron: an adaptive classifier for data streams. Pattern Recogn 44(1):78–96
Qiao JF, Han HG (2010) A repair algorithm for RBF neural network and its application to chemical oxygen demand modeling. Int J Neural Syst 20(1):63–74
Qiao J, Zhang Z, Bo Y (2014) An online self-adaptive modular neural network for time-varying systems. Neurocomputing 125:7–16
Qiao J, Li F, Han H, Li W (2016) Constructive algorithm for fully connected cascade feedforward neural networks. Neurocomputing 182:154–164
Qi M, Zhang GP (2001) An investigation of model selection criteria for neural network time series forecasting. Eur J Oper Res 132:666–680
Reitermanová Z (2008) Feedforward neural networks architecture optimization and knowledge extraction. In: Proceedings of week of doctoral students (WDS 2008), vol Part I, pp 159–164
Robins A (2004) Sequential learning in neural networks: a review and a discussion of pseudorehearsal based methods. Intell Data Anal 8(3):301–322
Rosenblatt F (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol Rev 65(6):386–408
Rumelhart DE, Hinton GE, Williams RJ (1986) Learning representations of back-propagation errors. Nature 323:533–536
Scarselli F, Tsoi AC (1998) Universal approximation using feedforward neural networks a surver of some existing methods and some new results. Neural Netw 11(1):15–37
Shao HM, Zheng GF (2011) Boundedness and convergence of online gradient method with penalty and momentum. Neurocomputing 74:765–770
Sharma SK, Chandra P (2010) Constructive neural networks: a review. Int J Eng Sci Technol 2(12):7847–7855

Subirats JL, Franco L, Jerez JM (2012) C-mantec: a novel constructive neural network algorithm incorporating competition between neurons. Neural Netw 26:131–140
Teoh EJ, Tan KC, Xiang C (2006) Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. IEEE Trans Neural Netw 17(6):1623–1629
Thomas P, Suhner MC (2015) A new multilayer perceptron pruning algorithm for classification and regression applications. Neural Process Lett 42(2):437–458
Vapnik V (1998) Statistical learning theory. Wiley, New York
Wang C, Hill DJ (2006) Learning from neural control. IEEE Trans Neural Netw 17(1):30–46
Wang J, Yang G, Liu S, Zurada JM (2015a) Convergence analysis of multilayer feedforward networks trained with penalty terms: a review. J Appl Comput Sci Methods 7(2):89–103
Wang J-H, Wang H-Y, Chen Y-L, Liu C-M (2015b) A constructive algorithm for unsupervised learning with incremental neural network. J Appl Res Technol 13:188–196
Widmer G, Kubat M (1996) Learning in the presence of concept drift and hidden contexts. Mach Learn 23:69–101
Widrow E, Hoff ME (1960) Adaptive switching circuits. In: Proceedings of IRE WESCON convention, pp 96–104
Wu W, Fan QW, Zurada JM, Wang J, Yang DK, Liu Y (2014) Batch gradient method with smoothing regularization for training of feedforward neural networks. Neural Netw 50:72–78
Xu J, Ho DWC (2006) A new training and pruning algorithm based on node dependence and jacobian rank deficiency. Neurocomputing 70(1–3):544–558
Yamakawa T, Uchino E, Miki T, Kusabagi H (1992) A neofuzzy neuron and its applications to system identification and predictions to system behavior. Proc Int Conf Fuzzy Logic Neural Netw 1:477–484
Ye Y, Squartini S, Piazza F (2013) Online sequential extreme learning machine in nonstationary environments. Neurocomputing 116:94–101
Yoan M, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A (2010) OP-ELM: optimally pruned extreme learning machine. IEEE Trans Neural Netw 21(1):158–162
Yu X, Chen QF (2012) Convergence of gradient method with penalty for ridge polynomial neural network. Neurocomputing 97:405–409
Zeng W, Wang C (2015) Classification of neurodegenerative diseases using gait dynamics via deterministic learning. Inf Sci 317(C):246–258
Zeng W, Wang C, Yang F (2014) Silhouette-based gait recognition via deterministic learning. Pattern Recogn 47(11):3568–3584
Zeng W, Wang Q, Liu F, Wang Y (2016) Learning from adaptive neural network output feedback control of a unicycle-type mobile robot. ISA Trans 61:337–347
Zhang HS, Wu W, Liu F, Yao MC (2009) Boundedness and convergence of online gadient method with penalty for feedforward neural networks. IEEE Trans Neural Netw 20(6):1050–1054
Zhang R, Lan Y, Huang GB, Xu ZB (2012) Universal approximation of extreme learning machine with adaptive growth of hidden nodes. IEEE Trans Neural Netw Learn Syst 23(2):365–371