International Journal of Hybrid Information Technology
Vol. 9, No.6 (2016), pp. 303-316
http://dx.doi.org/10.14257/ijhit.2016.9.6.27

A Novel Adaptive Architecture Pruning Algorithm for Madalines

Sai Ji[1], Ping Yang[1], Shuiming Zhong[1], Jin Wang[2] and Jeong-Uk Kim[3]

[1]*Jiangsu Engineering Center of Network Monitoring, Nanjing University of Information Science and Technology, Nanjing 210044，China*
[2]*School of Information Engineering, Yangzhou University, Yangzhou 225009, China*
[3] *Department of Energy Grid, Sangmyung University, Seoul, Korea*

Abstract

Nowadays, the big success of deep learning makes artificial neural network becoming a hot topic once again, and the size of neural networks' structure is a key visual cue for structured learning. The greater network may get the study task done well, while it may increase network computation overhead easier and cost more. Hence, network construction is an important issue, as well as a difficult problem. In this paper, we proposed a novel sensitivity-based adaptive architecture pruning algorithm for Madalines. The algorithm establishes a pruning measure based on the network sensitivity to its structure variation and a minimal disturbance principle. The measure can be used to evaluate the performance loss due to its structure changes more or less. And the loss can be compensated by relearning. Thus, the new adaptive pruning mechanism is developed with measuring, pruning, and compensating. The simulation experimental results based on some benchmark data demonstrate that the pruning measure is rationality and the new algorithm is effective.

Keywords: *Adaline; Madalines; sensitivity; network construction; pruning algorithm*

1. Introduction

In terms of architecture's function, neural network realizes an input-output mapping that based on the certain architecture and given weight. Therefore, the network construction and weight setting of neural network are hot topics in the research of neural network, and have drawn more and more researchers' attention [1, 2].

However, what is a proper architecture of a neural network for solving a given problem? The answer to this question is not easy. According to some achievements [3~5] of construction, there are some broad upper limit architectures, though those are not very significant to network construction. As a certain problem, if the network architecture is too small, it may cost less in both implementation and computation, while it may learn very slow or be not able to learn at all. When the network architecture is too big, it may be trained quickly and fit training data accurately, while it may cost more in implementation and computation and have bad performance in generalization [6]. Hence, the aim of network construction is to find a network with small but reasonable architecture and be able to learn well.

Researchers from different angles to explore network construction with different models for the long time, and there are some relevant techniques and methods have published [6~12]. At present, the network construction methods which are recognized by theorists and taken in engineering applications can be broadly divided into two categories, one is constructive algorithm [6,11] and the other is pruning algorithm [7,12]. The latter is more common in literatures. Pruning algorithm is a method that pruning neurons in hidden layer gradually based on a neural network with large size, to make the architecture of network smaller, until the task cannot be satisfied.

This paper discusses a new adaptive architecture pruning algorithm that based on

ISSN: 1738-9968 IJHIT

Madalines. A Madaline is a discrete feedforward neural network with hard limiting function and supervised learning mechanism, and compared with traditional continuous feedforward neural networks such as multilayer perceptron (MLP), the discrete character of Madalines have some advantages. On the one hand, Madalines are more suitable for handling many inherently discrete tasks which continuous techniques cannot handle, such as signal processing, logic operation, pattern classification and so on. On the other hand, Madalines are more conducive to the hardware implementation by the VLSI technology. Due to the hard limiting function is indifferentiable, many existing mature algorithms about architecture pruning based on continuous functions like MLP networks are not applicable to Madalines directly. Hence, it is necessary to explore new techniques to meet Madalines' discrete features.

As can be seen from the literature [1, 2, 7, 12], the topic of architecture pruning has been concerned by many researchers. Comparing those different methods from the surveys, we find that their main ideas underlying are almost the same. They all try to establish a reasonable relevance measure and hope to minimize the impact on the networks after pruning. Among those approaches, the methods based on sensitivity are commonly, which generally estimate the sensitivity of an objective function to achieve, such as the sensitivity for training errors [12, 13], for testing errors [14], or for the variation on output caused by its special parameters' disturbance [15]. Obviously, the sensitivity of former two is a local concept, they can only reflect local variations on output of the network, however, the variation is not only based on the training data or testing data, but also on the whole input space, which is a global concept. Therefore, the former two methods have limitations. Also, the last method need the specified parameters (such as weight or input) of the network be disturbed beforehand artificially or randomly, and then measure the importance of a neuron or neurons by calculating the sensitivity of the network's output. There is a bias error in this method and it cannot reflect the real impact on the network. In a word, their methods are more indirect and it is difficult to obtain theoretical support for correctness.

How to measure the variation on network's output caused by architecture changing more realistically, as well as design a suitable pruning algorithm for Madalines based on this measurement are major problem that need to be solved. This paper discusses the measurement form Madaline's sensitivity, which is a new perspective.

The aim of sensitivity researching is to explore how the variation on network's parameters (weight, input, architecture, *etc.*) affects the output of the network. In recent years, many achievements about sensitivity published [16~23]. Researchers employ different techniques from different view to explore sensitivity based on different models, solved some practical problems with sensitivity, such as network learning [19, 20], stability measurement [21,22] and network pruning [23] so on. However, the existing sensitivity almost all try to explore the variation on output caused by the variations on weight or input, while the sensitivity computation based on network's architecture is rarely seen from relevant conferences.

This paper measures the performance loss based on the existing theory of Madalines sensitivity computation. By calculating the variation on output caused by architecture pruning for Madalines, which is more substantive and can further conduct neurons pruning in the network. The main contribution of this paper is designed an adaptive architecture pruning algorithm for Madalines based on sensitivity. The algorithm can find the target network with smaller architecture adaptively, and optimize the construction of Madalines and reduce the computation of the network and the cost of hardware implementation effectively. The pruning algorithm has the following characteristics and innovations: (1) computing the variation on output of network caused by architecture changing directly, the results are based on the whole input space which is more practical; (2) creating the sensitivity measure follows the idea of minimal disturbance principle, which is more reasonable; (3) establishing the compensation mechanism to reduce the

performance loss of network caused by pruning network's architecture, which enhances the effectiveness.

The rest of this paper is organized as follows. In the next section, some preparation techniques are briefly described, including the Madalines model and notations, definition and Madalines' sensitivity computation based on weight disturbance, and Madalines' learning principles. Then, the Madalines' sensitivity definition and computation with variation on Madalines' architecture is deduced in Section 3. In Section 4, the pruning measurement is established based on sensitivity. Section 5 is the adaptive architecture pruning algorithm for Madalines based on sensitivity. Next, in Section 6, experimental verifications are given, which validated the rationality and validity of the new pruning algorithm. Finally, concludes this paper and discusses the future work on network construction in the last section.

2. Preliminaries

This section we briefly introduce some necessary techniques like the Madalines model, sensitivity computation of Madalines [16, 17] and sensitivity-based adaptive learning rules [20] (*SBALR*) for Madalines, for details please refer to the relevant conferences.

2.1. Madalines Model

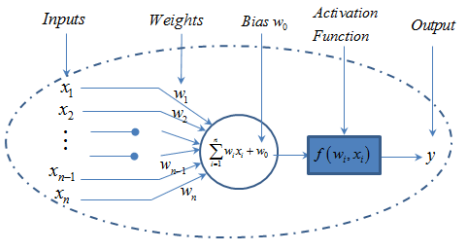


Figure 1. Adaline Model

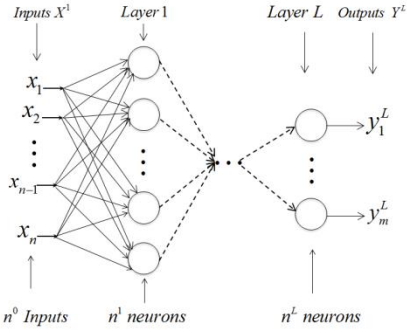


Figure 2. Madalines Model

A Madaline is a kind of discrete feedforward multilayer neural network and consists of a set of Adalines, it adopts a supervised learning mechanism. Adaline is the smallest and simplest unit of a Madaline, its activation function is hard limiting function which caused the input and output of Adaline with two values (+1 and -1).

Figure 1 is an Adaline model. Where $X=\left(x_1,x_2,\cdots,x_n\right)^T\in\left\{-1,1\right\}^n$ is the input, and $W=\left(w_1,w_2,\cdots,w_n\right)^T\in R^n$ is the weight correspond to its input, $w_0\in R$ is a bias, and $y\in\left\{-1,1\right\}$ is the output, and $f\left(\mathrm{g}\right)$ is its activation function, the hard limiting function express as formula (1).

$$y=f\left(\sum_{i=1}^{n}x_iw_i+w_0\right)=\begin{cases}-1, & \sum_{i=1}^{n}x_iw_i+w_0<0;\\ +1, & \sum_{i=1}^{n}x_iw_i+w_0\geq 0.\end{cases} \tag{1}$$

For convenience, in this paper, bias w_0 is seen as an element of Adaline's weight, and let its corresponding input element as $x_0\left(\equiv 1\right)$. In this way, the weight and input of Adaline can be further written as $W=\left(w_0,w_1,w_2,\cdots,w_n\right)^T\in R^{n+1}$ and $X=\left(x_0,x_1,x_2,\cdots,x_n\right)^T\in\left\{-1,1\right\}^{n+1}$.

Figure 2 is a Madaline model. Seeing from front to back, a Madaline is consisted of input layer, hidden layers and output layer. For discussion, we let $n^0 - n^1 - ... - n^L$ represents a Madaline with certain architecture. Where $n^l (1 \le l \le L)$ not only stands for a layer but also indicates the number of Adalines in the layer, while n^0 is an exception, it represents the dimension of Madaline's input, and n^L represents the output layer. In addition, X^l and $Y^l \left(1 \le l \le L\right)$ represent the input and output of the *l*th layer respectively, and there is $Y^{k-1} = X^k \left(2 \le k \le L\right)$, which means the output of the (*k-1*)th layer is the input of the *k*th layer. In particularly, X^1 is the input of the first layer and it is also the input of a Madaline, and the output of the last layer (the *L*th layer) is also the output of a Madaline.

According to relevant studies, for a single hidden layer network, as long as the number of neurons in hidden layer is more enough, the network can realize all the mapping relationship. Hence, the following discussions only focus on a Madaline with a single hidden layer.

2.2. Madalines Sensitivity

The sensitivity is to research the dependency relationship between networks' output variation and its parameter disturbance, as well as the existing methods mostly focus on the weight disturbance. This paper will take the existing methods as the foundations to further discuss the sensitivity computation with the architecture variation.

2.2.1. Adaline Sensitivity:

Definition 1. For an Adaline's input is $X \in \{-1,1\}^{n+1}$, the given weight is $W \in R^{n+1}$, and the corresponding variation on weight is $\Delta W \in R^{n+1}$, the sensitivity is defined as the probability of the Adaline's output inversion due to the variation on weight for all input points, which can be expressed as

$$s(\Delta W,\ \Delta X) = P\left(f\left(W^T X\right) \ne f\left((W + \Delta W)^T (X + \Delta X)\right)\right) = \frac{V_{\text{var}}}{V_n}, \quad (2)$$

where V_n is the number of all input points, and V_{var} is the number of the input points that cause the Adaline's output changed due to the variation on weight.

Because of the two value (-1 or 1) characteristic of Adalines' input and output, the sensitivity computation for Adaline based on the variation on input can converted into the variation on weight. For any element $x_i \left(1 \le i \le n\right)$ of the input, the variation on input can make $x_i^{'} = -x_i$, and there is the following equivalent transformation relation:

$$x_i^{'} w_i = (-x_i) w_i = x_i (-w_i) = x_i w_i^{'}, \quad (3)$$

where $w_i^{'} = -w_i = w_i - 2w_i$, and there is $\Delta w_i = -2w_i$.

The study [21] shows that the sensitivity of Adaline can be approximately computed as the following:

$$s(\Delta W) = \frac{\arccos\left(\left(W^T W^{'}\right)/\left(|W||W^{'}|\right)\right)}{\pi} \approx \frac{\left(|\Delta W|/|W|\right)}{\pi} \ for\ |\Delta W| = |W|, \quad (4)$$

where W, ΔW and $W^{'}$ represent the original weight, the variation on weight and the varied weight, respectively.

2.2.2. Madaline Sensitivity:

A Madaline is composed of layers, and a layer is composed of Adalines, with the variation on weight, definitions of layers and Madalines are expressed as the following:

Definition 2. For a Madaline's input is $X \in \{-1,1\}^{n+1}$, the given weight is $W_i^l \in R^{n^{l-1}+1}$, and the corresponding variation on weight is $\Delta W_i^l \in R^{n^{l-1}+1}$, where $\left(0 \le i \le n^l, 1 \le l \le L\right)$, the sensitivity of a layer is a vector and defined as all the neurons' sensitivity in this layer, which is expressed as

$$S^l = \left(s_1^l, s_2^l, \mathrm{L}\ s_{n^l}^l\right)^T \tag{5}$$

Definition 3. The sensitivity of a Madaline is the sensitivity of the output layer's sensitivity, which is expressed as

$$S_{net} = S^L = \left(s_1^L, s_2^L, \mathrm{L}\ s_{n^L}^L\right)^T \tag{6}$$

2.2.3. Sensitivity-Based Adaptive Learning Rules:

Different from the well-known BP algorithm [25], the learning rules on Madaline are not so mature, and this also one of the motivations of this study.

SBALR (Sensitivity-Based Adaptive Learning Rules) [20] is a weight adaptive learning algorithm for Madalines proposed based on perceptron learning rules [26, 27]. The algorithm mainly follows three principles: minimal disturbance principle, the benefit principle and the task allocation principle, and three learning rules are designed based on the three principles, as well as the most important learning rule is weight adaptive rule, which can be expressed as:

$$W' = \begin{cases} W + d\min\left(a_1, b_1\right)X & \left(\textit{for a hidden} - \textit{layer neuron}\right) \\ W + d\min\left(a_2, b_2\right)X & \left(\textit{for an output} - \textit{layer neuron}\right) \end{cases}, \tag{7}$$

where W and W' represent the weight before and after adjustment, X is the current input to the Adaline needs to adjust, $d\left(\in\{-1,+1\}\right)$ is the corresponding desire output of the Adaline, and parameters a_1, a_2, b_1, b_2 and can be further expressed as:

$$\begin{cases} \dfrac{\left|\left(W_i^1\right)^T X^1\right|}{n^1} @ a_1 & \left(\textit{for a hidden} - \textit{layer neuron}\right) \\ \dfrac{\left|\left(W_j^2\right)^T X^2\right|}{n^2} @ a_2 & \left(\textit{for an output} - \textit{layer neuron}\right) \end{cases}, \tag{8}$$

$$\begin{cases} \dfrac{1}{2}\pi^2 \left|W_i^1\right| \sqrt{\dfrac{n^1+1}{n^0+1}} \max\_ s_{net} @ b_1 & \left(\textit{for a hidden} - \textit{layer neuron}\right) \\ \dfrac{\pi\left|W_j^2\right|}{\sqrt{n^1+1}} \max\_ s_{net} @ b_2 & \left(\textit{for an output} - \textit{layer neuron}\right) \end{cases}, \tag{9}$$

where $\max\_ s_{net}$ is the parameter used to define the upper limit value of the sensitivity of the network caused by parameters of Madalines disturbance before training.

For more detailed introduction of the *SBALR* learning algorithm, please refer to [19].

3. Sensitivity of Madalines Based on Architecture

3.1. Sensitivity Definition Based on Architecture

On the one hand, the dimension of input and the number of output layer's neurons is determined by the specific learning tasks in a neural network, therefore the architecture pruning is only for hidden layer. This section discusses the influence of network's output that caused by pruning Adalines in hidden layer.

Definition 4. For a Madaline, it is a network with the architecture of $n^0 - n^1 - \mathrm{L} - n^L$, if the *j*th $(1 \le j \le n^i)$ Adaline in the *i*th $(1 \le l \le L)$ layer which marked as $node_j^i$ is pruned, the sensitivity of Madaline's architecture is defined as for any input points, the probability of the Madaline's output inversion because of the architecture pruning, and expressed as $s_{net}\left(-node_j^i\right)$.

According to formula (6), $s_{net}\left(-node_j^i\right)$ can be expressed as:

$$s_{net}\left(-node_j^i\right) = S^L = \left(s_1^L, s_2^L, \mathrm{L}\ s_{n^L}^L\right)^T. \tag{10}$$

It is obviously that the sensitivity of Madalines expressed in formula (10) is a vector, for convenience, formula (10) can be further quantified as the following:

$$S_{net} = \frac{\sum_{i=1}^{n^L} s_i^L \cdot V_n}{n^L \cdot V_n} = \frac{1}{n^L}\sum_{i=1}^{n^L} s_i^L, \tag{11}$$

where $s_i^L\left(1 \le i \le n^L\right)$ represents the *i*th Adaline's sensitivity of the output layer, and V_n is the number of all input points.

According to formula (11), the sensitivity of Madalines is equal to the average of all Adalines' sensitivity in output layer.

3.2. Sensitivity Computation based on Architecture

In a Madaline, if the *j*th $(1 \le j \le n^1)$ Adaline in hidden layer was pruned, then all the Adalines in output layer would loss the *j*th input element $x_{k,j}^2\left(1 \le k \le n^2\right)$, which would make the corresponding weight equals to 0, and the whole result of $\left(W_k^2\right)^T X_k^2$ changes from $\sum_{i=0}^{n^1} w_{k,i}^2 x_{k,i}^2$ to $\sum_{\substack{i=0 \\ i \ne j}}^{n^1} w_{k,i}^2 x_{k,i}^2$, which means,

$$\sum_{i=0}^{n^1} w_{k,i}^2 x_{k,i}^2 \xrightarrow[\text{in hidden layer}]{\text{prune the } j\text{th Adaline}} \sum_{\substack{i=0 \\ i \ne j}}^{n^1} w_{k,i}^2 x_{k,i}^2. \tag{12}$$

Since there is

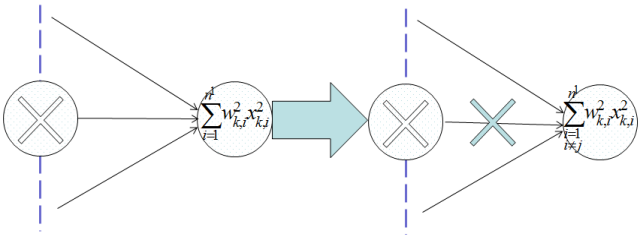
$$\sum_{\substack{i=0 \\ i \ne j}}^{n^1} w_{k,i}^2 x_{k,i}^2 = \sum_{\substack{i=0 \\ i \ne j}}^{n^1} w_{k,i}^2 x_{k,i}^2 + \left(w_{k,j}^2 - w_{k,j}^2\right) x_{k,j}^2, \tag{13}$$

where $1 \le j \le n^1, 1 \le k \le n^2$.

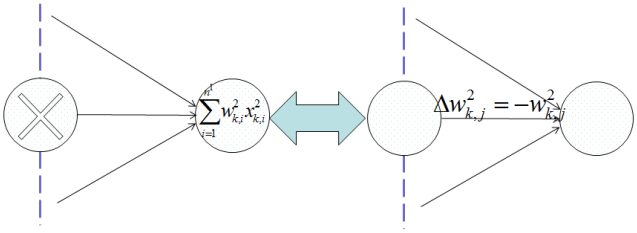
From formula (13), it can be seen that if pruning the *j*th Adaline in hidden layer, all the Adalines in output layer will loss the *j*th element of input. It can make an equivalent transform that the variation on weight cause the weight equals to 0, therefore there is $\Delta w_{k,j}^2 = -w_{k,j}^2$.

This equivalent transform can be further shown as the following structure chart (Figure 3). In Figure 3 (a), the real situation of pruning an Adaline in hidden layer, which causes the dimension of the input to output layer reducing one, and the Figure 3 (b) is the

equivalence situation, which vividly shows that the weight to output layer disturbed, and there is $\Delta w_{k,j}^2 = -w_{k,j}^2$.



(a) Real situation



(b) Equivalence situation

Figure 3. The Effect of Hidden Layer Nodes Reduction on the Output Layer Nodes

In the situation of pruning the *j*th Adaline in hidden layer, the sensitivity computation of any Adaline in output can transform to a disturbance in weight, which names:

$$s_k^2\left(-node_j^1\right) = s_k^2\left(\Delta W_k^2\right), \Delta w_{k,i}^2 = \begin{cases} 0 & i \neq j \\ -w_{k,i}^2 & i = j \end{cases}, \tag{14}$$

where $\Delta w_{k,i}^2 \left(0 \le i \le n^1\right)$ is the *i*th element of ΔW_k^2.

Combined with formula (13), the formula (14) can be further written as:

$$\begin{aligned} s_k^2\left(-node_j^1\right) = s_k^2\left(\Delta W_k^2\right) &= \frac{1}{\pi}\arccos\left(\left(\left(W_k^2\right)^T W_k^{2\prime}\right) \Big/ \left(\left|W_k^2\right|\left|W_k^{2\prime}\right|\right)\right) \\ &= \frac{1}{\pi}\arccos\left(\frac{\sum_{\substack{i=0 \\ i\neq j}}^{n^1}\left(w_{k,i}^2\right)^2}{\sqrt{\sum_{i=0}^{n^1}\left(w_{k,i}^2\right)^2}\sqrt{\sum_{\substack{i=0 \\ i\neq j}}^{n^1}\left(w_{k,i}^2\right)^2}}\right) = \frac{1}{\pi}\arccos\left(\frac{\sqrt{\sum_{\substack{i=0 \\ i\neq j}}^{n^1}\left(w_{k,i}^2\right)^2}}{\sqrt{\sum_{i=0}^{n^1}\left(w_{k,i}^2\right)^2}}\right), \\ &= \frac{1}{\pi}\arccos\left(\frac{\left|W_k^{2\prime}\right|}{\left|W_k^2\right|}\right) \approx \frac{1}{\pi}\frac{\left|\Delta W_k^2\right|}{\left|W_k^2\right|} \qquad for\ \left|\Delta W_k^2\right| = \left|W_k^2\right| \end{aligned} \tag{15}$$

where $\Delta W_k^2 = \left(\Delta w_{k,0}^2, \Delta w_{k,1}^2, \mathrm{L}\ , \Delta w_{k,n^1}^2\right)^T = \left(0, \mathrm{L}\ , -w_{k,j}^2\ \mathrm{L}\ \right)^T$.

According to formula (15), all the Adalines' sensitivity in output layer caused by pruning the *j*th Adaline in hidden layer can figure out, and then like formula (11), the sensitivity of the Madaline can figure out too, that is

$$s_{net}\left(-node_j^1\right) = \frac{1}{n^2}\sum_{i=1}^{n^2} s_i^2. \tag{16}$$

In fact, the sensitivity computation formulae of the Madaline pruning an Adaline derived above can also be generalized to prune Adalines.

4. Pruning Measure Based on Sensitivity

To seek a target network is the direct motivation of exploring network architecture pruning, as well as the target network can meet the task learning with simplified architecture by pruning Adalines in hidden layer. However, among all the neurons in hidden layer, which one should be pruned and what is the basis for selection must be given reasonable answers when design of pruning algorithm.

For a trained network, it is obviously that pruning Adalines in hidden layer will cause variation on the output of the Madaline, it will also change the performance of the existing network. However, how to measure the performance qualitatively. It is difficult to measure the learning and generalization of the network getting better or worse, though, quantitative calculation for this change in the Madaline is feasible. According to the definition of sensitivity and the result of formula (16), sensitivity can more directly reflect the degree of variation on Madaline's output caused by pruning Madaline's architecture, as the following

$$netoutput\ \ change = s_{net}\left(-node_j^i\right), \tag{17}$$

where $-node_j^i$ represents pruning the jth $(1 \le j \le n^i)$ Adaline in ith $(1 \le i \le L)$ the layer.

Since a successful pruning approach is the one that can make the pruned Madaline recover the convergence to the state before pruning, therefore, selecting which Adaline to prune should be analyzed from the view of learning. The existing Madalines learning mechanisms all follow an important principle, which is namely minimal disturbance principle [19, 20, 23]. Upon training a Madaline, it should meet reducing the output error or tend to reduce the output error for the current training sample as well as the weight adjustment should reduce break the mapping relationship established by other training samples. Intuitively, the smaller variation on a Madaline's output, the easier to recover to the original performance by relearning; while the greater variation on a Madaline's output, the more difficult to recover to the original performance by relearning. The influence of the network's output and performance caused by architecture pruning is not what we want, while it is objective existence. Hence, we hope the variation on output as small as possible.

Based on the above discussion, the Adaline pruning approach can be described as: among the multiple Adalines in the hidden layer, pruning which one should be based on the rule that the smaller variation on the Madaline's output after pruning should be given priority. The selection strategy further evolved into pruning selection measure, which can be denoted as:

$$pruning\ selection\ measure = \min\left(s_{net}\left(-node_1^1\right), s_{net}\left(-node_2^1\right), \mathrm{L}\ , s_{net}\left(-node_{n^1}^1\right)\right), \tag{18}$$

where $-node_j^i$ represents pruning the jth $(1 \le j \le n^i)$ Adaline in ith $(1 \le i \le L)$ the layer.

5. The Pruning Algorithm

In Section 3 and Section 4, the quantitative computation of the variation on Madaline's output (sensitivity computation) as well as the pruning strategy and the pruning measure selection are discussed. Combined with the existing *SBALR* [19] for Madalines and the actual situation, this section mainly designs the adaptive architecture pruning algorithm based on sensitivity. The adaptive architecture pruning algorithm can be briefly described as the Figure 4.

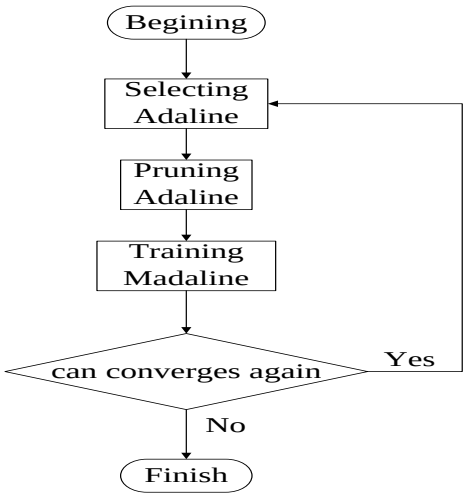


Figure 4. The Adaptive Architecture Pruning Algorithm

However, the adaptive learning algorithm has inherent limitations. For a certain task, there is a reasonable network size in theory while the adaptive learning mechanism cannot ensure to find it. This is because when pruning a neuron failed, the mechanism will stop pruning, which may miss the opportunity to look for a smaller construction. In order to find a Madaline with smaller size, we considered trying more times for pruning an Adaline. The times of trying to prune an Adaline generally can be set as 3. For a given task, after continuous three times trying failed, then the current architecture of Madaline is considered as the smallest one.

Hence, the specific adaptive architecture pruning algorithm for Madalines based on sensitivity can be further described as following:

Algorithm 1: A Sensitivity-Based Adaptive Architecture Pruning Algorithm for Madalines

Input: a set of training data, testing data; the number of iterative in each training; training expected accuracy; the maximum number (K) of continuous failures when pruning an Adaline.

(1) Initializing a Madaline with a larger size according to a given learning task;

(2) Training a Madaline with *SBALR*;

(3) If the Madaline after training cannot reach the expected training accuracy, then initialize a Madaline with larger size, and return to step (2);

(4) Save the current Madaline;

(5) $For\ \ i=1\ \ to\left(n^1-2\right)$, (where n^1 is the number of Adalines in hidden layer)

1) According to formula (16), calculate the sensitivity $\left(=s_{net}\left(-node_s^1\right)\left(1\le s\le n^1\right)\right)$ of Madaline after pruning each Adaline in hidden layer, according to the sensitivity from small to large, sort the Adalines, and make the first K Adalines in ascending order queue Q;

2) Initializing the continuous times for pruning an Adaline $NoFlag\leftarrow 0$;

3) $For\ \ j=1\ \ to\ \ K$;

a) According to formula (18), choose the *j*th Adaline in Q queue, and prune it;

b) Using *SBALR* learning algorithm for the retraining the Madaline after pruning an Adaline;

c) If the Madaline after retraining cannot achieve accuracy requirement, then take the trained Madaline replace the current Madaline, and save the trained Madaline, and

jump out of the loop;

d) $NoFlag \leftarrow NoFlag + 1$;

E*ndfor*

4) If there is $NoFlag == K$, then jump out of the loop, and stop pruning;

E*ndfor*

Output: save the current Madaline as the target Madaline.

6. Experimental Verifications

Table 1. Data Sets Used In the Experiments

Data set	Attribute	Class	Set size
7BIT-PARITY	7	2	128
AND-XOR	2	2	4
MONKS-1	10	2	556
BALANC-SCALE	12	3	625

This section presents some experiments which mainly verify two points. One is to verify the reasonable of the pruning measure and strategy; another is to verify the effectiveness of the sensitivity-based adaptive architecture pruning algorithm for Madalines. And these experiments are based on some public data sets, which are shown in Table 1.

In order to verify the reasonable of the pruning measure and strategy, the initial Madaline with the architecture of 7-8-1 was selected, as well as the target Madaline should with the architecture of 7-7-1. In the experiments, the problem of the 7 bit parity (one parity problem) was solved. Firstly in the case of each Adaline in hidden layer was selected to prune, and then the sensitivity caused by pruning each Adaline were extracted, respectively. The time index value of retraining the Madaline to reach the convergence state after pruning is called the times of Adaline adjustment, which is used to measure the learning efficiency of learning algorithms [20]. Under the condition of initializing Madaline's weight randomly, 2 simulation experiments were carried out and the experimental results are shown in Table 2 (a) (b).

Table 2 The efficiency of network retraining under different hidden layer nodes

Initial Madaline	No. of Adaline in hidden layer	sensitivity	The times of node adjustment after relearning
7-8-1	7	0.031669	24
	4	0.057620	90
	1	0.087699	1590
	2	0.092457	2719
	8	0.118963	36709
	5	0.137162	195
	6	0.139436	207
	3	0.167615	1455

(a)

Initial Madaline	No. of Adaline in hidden layer	sensitivity	The times of node adjustment after relearning
7-8-1	2	0.020953	12
	3	0.047819	1080004
	7	0.055710	143
	1	0.058946	913
	3	0.060605	74098
	4	0.072511	1429
	5	0.201693	2194930
	8	0.206000	2368098

(b)

It can be seen from table 2, the Madaline's sensitivity after pruning an Adaline in hidden layer is in direct proportion to the time of retraining the Madaline to reach the convergence state after pruning. It means that the Adaline caused the smaller sensitivity of Madaline should be pruned priority, since it can return to the performance after pruning quicker. The proposed pruning measure and strategy is reasonable that is by verified by the experiments, as well as it is feasible.

In order to verify the effectiveness of the sensitivity-based adaptive architecture pruning algorithm for Madalines, the following experiments are carried out. In the experiments, the problems like AND-XOR (A logical operation problem), MONKS-1 and BLANCE-SCALE from the UCI repository are solved with *SBALR* algorithm and pruning algorithm in this paper. The results are shown table 3. In order to guarantee the feasible of the experiments and the validity of experimental results, the number of training iterations is 200, 2000 and 60000, respectively, and three times for continuous pruning an Adaline are allowed. Each result comes from the average of 100 runs. The training results (convergence rate of network which is the percentage of successful training samples for a trained Madaline) and training efficiency (the actual number of iterations for convergence and the number of Adaline adjustment for convergence) are shown in table 3.

Table 3. Comparison of the Effectiveness and Efficiency of yhe Network Training with the Pruning Algorithm and the (*SBALR*)

Data set	The pruning algorithm in this paper					*SBALR*			
	Initial	Target	conv. (%)	Iter.	Adju.	Target .	conv. (%)	Iter.	Adju.
AND-XOR	2-3-2	2-2-2	89	11	10	2-2-2	53	23	23
MONKS-1	10-4-1	10-3-1	98	180	698	10-3-1	85	388	1304
BALANC-SCALE	12-10-2	12-9-2	99	1703	9251	12-9-2	81	2266	10189
		12-8-2	97	1756	10351	12-8-2	80	2195	9567
		12-7-2	93	1131	6092	12-7-2	68	3494	13445
		12-6-2	92	2366	9934	12-6-2	71	5702	21829
		12-5-2	91	6871	28817	12-5-2	65	11533	56087
Initial represents Initial Madaline, Target represents Target Madaline, conv. represents convergence rate %, Iter. represents Iteration, adju. represents adjustments.									

As table 3 shows, pruning a Madaline with lager size by the algorithm discussed in this paper can get the higher convergence rate and better training effect, while training the target Madaline directly with *SBALR* algorithm gets the lower convergence rate. Also, the target Madaline return to the state before pruning needs less iteration and fewer times Adaline adjustment by the proposed algorithm, which proves the proposed algorithm has a very good inheritance to the performance of the network once again.

7. Conclusion

In this paper we proposed a new adaptive architecture pruning algorithm based on sensitivity. It can quantize the influence of Madaline's output caused by architecture pruning. The proposed algorithm can obtain a target network with simpler architecture and better performance automatically by adaptive searching, as well as inheritance is the biggest feature of it. The algorithm can calculate the variation on Madaline's output directly, which can further enrich the theoretical and technical of the construction of the Madaline. The characteristics of architecture optimization and performance inheritance will bring some inspiration to study of learning behavior, as well as whether learning and

construction of network can consider from the view of learning behavior. This will be the next topic we concerning in future.

Acknowledgements

This paper is a revised and expanded version of a paper entitled "A Sensitivity-Based Adaptive Architecture Pruning Algorithm for Madalines" presented at CST 2016 China, April 22-23. This work was supported by the National Natural Science Foundation of China (51505234, 51305211, 61300237, 61402234, 61402235, 61311140264) and the PAPD. It was also supported by the Industrial Strategic Technology Development Program (10041740) funded by the Ministry of Trade, Industry and Energy (MOTIE) Korea. Prof. Jeong-Uk Kim is the corresponding author.

References

[1] R. Reed, "Pruning algorithms-a survey", IEEE Transactions on Neural Networks, vol. 4, no. 5, **(1993)**, pp. 740-747.
[2] M. G. Augasta and T. Kathirvalavakumar, "Pruning algorithms of neural networks-a comparative study", Open Computer Science,vol. 3, no.3, **(2013)**, pp. 105-115.
[3] Z. Z. Zhang, X. M. Ma and Y. X. Yang, "Bounds on the number of hidden neurons in three-layer binary neural networks", Neural Networks, vol. 16, no. 7, **(2003)**, pp. 995–1002.
[4] B. Choi, J. H. Lee and D. H. Kim, "Solving local minima problem with large number of hidden nodes on two-layered feed-forward artificial neural networks", Neurocomputing, vol. 71, no. 16, **(2008)**, pp. 3640-3643.
[5] W. J. Puma-Villanueva, E. P. D. Santos and F. J. V. Zuben, "A constructive algorithm to synthesize arbitrarily connected feedforward neural networks", Neurocomputing, vol. 75, no. 1, **(2012)**, pp. 14-32.
[6] X. Z. Wang, Q. Y. Shao and Q. Miao, "Architecture selection for networks trained with extreme learning machine using localized generalization error model", Neurocomputing, vol. 102, no. 2, **(2013)**, pp. 3-9.
[7] M. R. Silvestre and L. L. Ling, "Pruning methods to MLP neural networks considering proportional apparent error rate for classification problems with unbalanced data", Measurement, vol. 56, no. 19, **(2014)**, pp. 88-94.
[8] M. Han and J. Yin, "The hidden neurons selection of the wavelet networks using support vector machines and ridge regression", Neurocomputing, vol. 72, no. 1-3, **(2008)**, pp. 471-479.
[9] T. Y. Kwok and D. Y. Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems", IEEE Transactions on Neural Networks, vol. 8, no. 3, pp. 630-645.
[10] J. H. Wang, H. Y. Wang and Y. L. Chen, "A constructive algorithm for unsupervised learning with incremental neural network", Journal of Applied Research and Technology, vol. 12, no. 2, **(2015)**, pp. 188–196.
[11] X. Wu, P. Rozycki and B. M. Wilamowski, "A Hybrid Constructive Algorithm for Single-Layer Feedforward Networks Learning", IEEE Transactions on Neural Networks & Learning Systems, vol. 26, no. 8, **(2014)**, pp. 1-1.
[12] K. Suzuki, I. Horiba and N. Sugie, "A Simple Neural Network Pruning Algorithm with Application to Filter Synthesis", Neural Processing Letters, vol. 13, no. 1, **(2001)**, pp. 43-53.
[13] J. Gorodkin, L. K. Hansen and A. Krogh, "A quantitative study of pruning by optimal brain damage", International Journal of Neural Systems, vol. 4, no. 2, **(1993)**, pp. 159-169.
[14] X. Q. Zeng and D. S.Yeung, "Hidden neuron pruning of multilayer perceptrons using a quantified sensitivity measure", Neurocomputing, vol. 69, no. 7-9, **(2006)**, pp. 825-837.
[15] A. P. Engelbrecht, "A new pruning heuristic based on variance analysis of sensitivity information", IEEE Transactions on Neural Networks, vol. 12, no. 6, **(2001)**, pp. 1386-1399.
[16] S. M. Zhong, X. Q. Zeng and H. Y. Liu, "Approximate computation of Madaline sensitivity based on discrete stochastic technique", Sciece China Information Sciences, vol. 53, no. 12, **(2010)**, pp. 2399-2414.
[17] L. h. Huang, X. Q. Zeng and S. M. Zhong, "Sensitivity study of Binary Feedforward Neural Networks", Neurocomputing, vol. 136, no. 1, **(2014)**, pp. 268-280.
[18] X. Q. Zeng, Y. F. Wang and K. Zhang, "Computation of Adalines' sensitivity to weight perturbation", IEEE Transactions on Neural Networks, vol. 17, no. 2, **(2006)**, pp. 515-519.
[19] D. S. Yeung, P. P. K. Chan and W. W. Y. Ng, "Radial Basis Function network learning using localized generalization error bound", Information Sciences,19, vol. 179, no. **(2009)**, pp. 3199-3217.
[20] Zhong Shuiming, Zeng Xiaoqin and Wu Shengli, "Sensitivity-Based Adaptive Learning Rules for Binary Feedforward Neural Networks", IEEE Transactions on Neural Networks and Learning Systems, vol. 23, no. 3, **(2012)**, pp. 480-491.
[21] J. L. Bernier, J. Ortega and E. Ros, "A Quantitative Study of Fault Tolerance, Noise Immunity, and

Generalization Ability of MLPs", Neural Computation, vol. 12, no. 12, **(2006)**, pp. 2941-2964.
[22] D. S. Yeung, W. W. Y. Ng and W. Defeng, "Localized generalization error model and its application to architecture selection for radial basis function neural network", IEEE Transactions on Neural Networks, vol. 18, no. 5, **(2007)**, pp. 1294-1305.
[23] X. Q. Zeng, J. Shao and Y. F. Wang, "A sensitivity-based approach for pruning architecture of Madalines", Neural Computing and Applications, vol.18, no. 8, **(2009)**, pp. 957-965.
[24] N. E. Cotter, "The Stone-Weierstrass theorem and its application to neural networks", IEEE Transactions on Neural Networks, vol. 1, no. 4, **(1990)**, pp. 290-295.
[25] D. Rumelhart and J. Mcclelland, "Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations", MIT Press, **(1987)**.
[26] B. Widrow and E. Walach, "Appendix G: Thirty Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation", Proceedings of the IEEE, vol. 78, no. 9, **(1990)**, pp. 1415-1442.
[27] F. Rosenblatt, "On the convergence of reinforcement procedures in simple perceptrons", Tech. Rep. VG-1196-G-4, Cornell Aeronautical Laboratories, **(1960)**.

Authors

Sai Ji, Associate Professor at College of computer & Software at Nanjing University of Information Science & Technology. His research interest covers wireless sensor networks, machine learning, and computational intelligence.

Ping Yang, M.S. candidate at College of computer & Software at Nanjing University of Information Science & Technology. Her main research interests include neural networks and machine learning.

Shuiming Zhong, Lecturer at College of computer & Software at Nanjing University of Information Science & Technology. His research interests involve artificial neural networks, machine learning, and pattern recognition.

Jin Wang received the B.S. and M.S. degree from Nanjing University of Posts and Telecommunications, China in 2002 and 2005, respectively. He received Ph.D. degree from Kyung Hee University Korea in 2010. Now, he is a professor in the School of Information Engineering, Yangzhou University. His research interests mainly include routing method and algorithm design, performance evaluation and optimization for wireless ad hoc and sensor networks. He is a member of the IEEE and ACM.

Jeong-Uk Kim received his B.S. degree in Control and Instrumentation Engineering from Seoul National University in 1987, M.S. and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology in 1989, and 1993, respectively. He is an associated professor at the SangMyung University in Seoul. His research interests include smart grid demand response, building automation system, and renewable energy.

