Artificial Neural Networks
T. Kohonen, K. Mäkisara, O. Simula and J. Kangas (Editors)

Focused Incremental Learning for Improved Generalization with Reduced Training Sets

Byoung-Tak ZHANG & Gerd VEENKER

Division I – Artificial Intelligence
Institute for Computer Science
University of Bonn, 5300 Bonn, Germany
Email: zhang,veenker@dbninf5.bitnet

Abstract

This paper presents an incremental learning procedure for improving generalization performance of multilayer feedforward networks. Whereas most existing algorithms try to reduce the size of the network to improve the likelihood of finding solutions of good generalization, our method constrains the search by incremental selection of training examples according to the estimated usefulness, called *interestingness*, of the example. It is shown in Boolean function tasks that the incremental algorithm achieves better generalization with smaller training sets than the nonincremental ones.

1 Introduction

Learning in neural networks can be regarded as a search process. The adjustable connection weights of the network constitute a search space, part of which is the solution space. The process of learning is a search for a point in the solution space that realizes the mappings in a training set. The difficulty in generalization, i.e. the abilities of networks to respond to novel inputs after learning, is that there are many different solution points that correctly classify the training examples but which fail to reasonably respond to new inputs. To achieve good generalization property, therefore, additional constraints should be imposed during the search for a solution.

One typical way of imposing constraints is to minimize the number of free parameters or the size of the network [6]. This can be done by adding a cost term that penalizes large size networks [2,3,5,7]. The basic idea behind this approach is that reducing the number of free parameters reduces the number of possible solutions and in doing so may eliminate some undesirable ones.

However, there are other ways of imposing constraints. Recent studies have shown that among various examples there are some specific patterns called *border* or *critical patterns* which improve generalization more than others [1,4]. Another study on human learning of binary categorization tasks has revealed that subject's biases of early learning stages may provide important constraints on later generalization [8].

The learning procedure proposed in this paper is an incremental algorithm which makes use of the latter points: it constrains the search by first focusing on critical patterns to provide better starting conditions for the next step of the search. The algorithm is described in Section 2. Section 3 and 4 contain the results on learning majority functions and an analysis of the algorithm based on the results, respectively. Some concluding remarks on the implications of this work are given in Section 5.

2 The Algorithm

The algorithm is summarized in the following. First, the network and training sets are initialized (step 1). Two sets of training examples are distinguished in the algorithm: an *exempel base* $\Phi = \{\varphi_i\}$ which waits to be used for training, and an *exemplar base* $\Psi = \{\psi_i\}$ which is used in the training of the network. In essence, the learning process is an iteration of an adaptation and a selection phase.

---

1. **Initialization**
   Initialize the parameters $\tau, \epsilon, \eta, \kappa, \alpha, \beta, d_\alpha, d_\beta$. Initialize the connection weights Ω of the n-h-m fully-connected feedforward network Γ. Initialize Ξ with the given test set. Set Φ_0 with the examples given by the teacher. Set $\Psi_0 \leftarrow \emptyset$. Set $s \leftarrow 1$. Select one example $\varphi_p \in \Phi$ randomly for each category of the examples and set $\Psi \leftarrow \Psi_0 \cup \{\varphi_p\}$, $\Phi \leftarrow \Phi_0 - \{\varphi_p\}$.

2. **Adaptation**

   (a) (Termination criterion) Compute λ by Equation 1. Set $t \leftarrow 0$.

   (b) (Focused BP) Set $t \leftarrow t + 1$. Set $E \leftarrow 0$. For each $\psi_p = (X_p, Y_p, e_p) \in \Psi$ do the followings: Compute $E_p = (Y_p - \Gamma(\Omega, X_p))^T (Y_p - \Gamma(\Omega, X_p))$. Set $E \leftarrow E + E_p$. Compute the interestingness $e_p = \frac{1}{m}\sqrt{E_p}$. Modify the connection weights $\Omega \leftarrow \Omega + \Delta\Omega$, where $\Delta\Omega = \{\Delta w_{ji}\}$ and Δw_{ji} is defined by Equation 2.

   (c) (Termination test) If $E > \lambda$, go to step 2b. Test the network with the test set Ξ and compute $G = 100 \cdot \frac{N_{correct}}{|\Xi|}$. If $G = 100$ or $\Phi = \emptyset$, then stop learning.

3. **Selection**

   (a) (Interestingness) Set $s \leftarrow s + 1$. For every $\varphi_p = (X_p, Y_p, e_p) \in \Phi$, reevaluate the interestingness e_p by Equation 3.

   (b) (Survival) Choose maximal κ examples $\varphi_p \in \Phi$ with $e_p \geq \beta$ for each category of the examples and set $\Psi \leftarrow \Psi \cup \{\varphi_p\}$, $\Phi \leftarrow \Phi - \{\varphi_p\}$.

   (c) (Expulsion) Choose $\varphi_q \in \Phi$ with $e_q < \alpha$ and set $\Phi \leftarrow \Phi - \{\varphi_q\}$. Set $\beta \leftarrow \beta - d_\beta$ and $\alpha \leftarrow \alpha + d_\alpha$. Go to step 2.

The focused incremental learning algorithm

In the adaptation phase (step 2), the connection weights of the network are updated using only the training patterns in the exemplar base. In this update, an on-line "attentive" variation of the backpropagation error correction algorithm [9], what we call a *focused BP*, is used. This connection weight adjustment is repeated until $E \leq \lambda$ is satisfied. The termination criterion λ is computed dynamically, depending on the current sizes of the network and the exemplar base, by

$$\lambda = \frac{1}{\tau} \cdot |\Psi| \cdot h(n + m). \tag{1}$$

The weight adjustment rate, ϵ_p, in the focused BP is not equal for all examples, but dependent on the *interestingness*, e_p, of the examples:

$$\Delta w_{ji}^{new} = \eta \Delta w_{ji}^{old} + \epsilon_p \delta_j y_i = \eta \Delta w_{ji}^{old} + \epsilon(1 + e_p)\delta_j y_i. \tag{2}$$

In the equation, the error signal δ_j is computed by $\delta_j = a_j(1 - a_j)(y_j - a_j)$ for $j \in \mathcal{O}$, $\delta_j = a_j(1 - a_j)\sum_{k \in \mathcal{O}} \delta_k w_{kj}$ for $j \in \mathcal{H}$, where a_j is the actual output of the j-th unit. The interestingness of an example p on the basis of the knowledge Ω is defined as

$$e_p = \frac{1}{m}\sqrt{(Y_p - \Gamma(\Omega, X_p))^T (Y_p - \Gamma(\Omega, X_p))} \tag{3}$$

where m, X_p, Y_p, and $\Gamma(\Omega, X_p)$ are the number of output units and the input, desired, and actual output vectors, respectively. The more interesting an example is, the higher the adjustment rate of the example. This ensures quicker adaptation of critical patterns to the existing knowledge structure $\Omega = \{w_{ji}\}$.

In the selection phase (step 3) the interestingness of the examples in Φ is computed. Among them for each category maximal κ examples with the interestingness $e_p > \beta$ are selected and are moved to the exemplar base (Survival). The uninteresting examples, those with $e_p < \alpha$, are eliminated from Φ (Expulsion). As a result, the set Φ stores examples with interestingness $\alpha \leq e_p < \beta$ for the current knowledge base Ω. Then, the network is trained again with the new exemplar base. This process of selection and adaptation is repeated until the expected value of the generalization performance, G, reaches 100% or the exempel base Φ contains no more examples. At the termination of learning, Ψ contains the examples actually used in training the network.

3 Results

The generalization capabilities of the selective incremental and the nonincremental (focused BP) algorithm were tested on the majority function: return 1 if more than half of the input units are 1. To compare the performances we used the following measures: $G(s) = 100 \cdot \frac{N_{correct}(s)}{|\Xi|}$, $D(s) = 100 \cdot \frac{|\Psi_s|}{|\Psi_0|+|\Phi_0|}$ and $T(s) = \sum_s |\Psi_s| \sum_1^{t(s)} h(n+m)$, where s and $t(s)$ are the selection and adaptation steps, respectively.

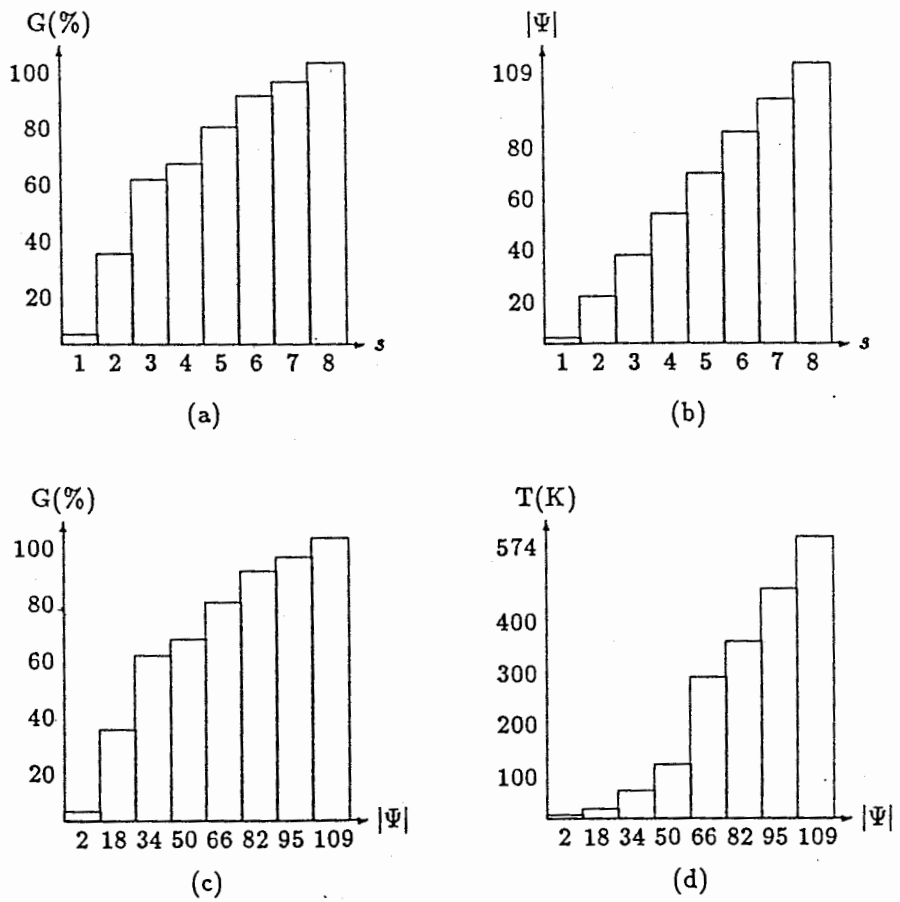

Figure 1: The relations among s, G, $|\Psi|$, and T in the course of an incremental learning.

G is a measure of generalization performance estimated by the test set. D is the size of the training set actually used in training the network. This measure indicates the ability of the learning algorithm to select useful patterns. T measures learning time in the total number of connection weight adjustments. It is important to note that T is not calculated in epochs because each epoch in the incremental learning uses different sizes of training sets. Figure 1 shows the relations among s, G, $|\Psi|$, and T in the course of an incremental learning of the 25-input majority function, given 400 random examples and with the parameter $\kappa = 8$. The learning trial arrived at 100% generalization with 109 training examples incrementally selected from 400 given examples. It made 574K connection weight adjustments in 8 selection steps. The generalization performance was estimated by a test set of 1000 random examples.

Tables 1-3 show the results for input size $n = 5, 15$ and 25, respectively. For each task, three different sizes of initial training sets, N, are used. In all experiments, networks with one hidden unit were used. The training and test sets are generated randomly and Ψ_1 contained two examples chosen randomly from Φ_0. The parameter values for all experiments were $\tau = 1000, \epsilon = 0.1, \eta = 0.9$, $\alpha = 0.0$ and $\beta = 0.3$, $d_\alpha = 0.0$ and $d_\beta = 0.01$.

N	Incremental			Nonincremental			Relative performance		
	$G_1(\%)$	$D_1(\%)$	T_1(K)	$G_2(\%)$	$D_2(\%)$	T_2(K)	G_1/G_2	D_1/D_2	T_1/T_2
15	88.4	87.9	13.6	84.0	100.0	11.3	1.05	0.88	1.20
20	98.8	59.0	14.0	98.2	100.0	12.8	1.01	0.59	1.09
25	100.0	41.6	15.0	100.0	100.0	13.2	1.00	0.42	1.14

Table 1: Majority function: $n = 5, \kappa = 1, |\Xi| = 32$

N	Incremental			Nonincremental			Relative performance		
	$G_1(\%)$	$D_1(\%)$	T_1(K)	$G_2(\%)$	$D_2(\%)$	T_2(K)	G_1/G_2	D_1/D_2	T_1/T_2
100	91.2	100.0	109.8	83.8	100.0	77.2	1.09	1.00	1.42
150	100.0	38.2	92.4	87.6	100.0	106.6	1.14	0.38	0.87
200	100.0	32.2	85.8	91.2	100.0	107.4	1.10	0.32	0.80

Table 2: Majority function: $n = 15, \kappa = 3, |\Xi| = 500$

N	Incremental			Nonincremental			Relative performance		
	$G_1(\%)$	$D_1(\%)$	T_1(K)	$G_2(\%)$	$D_2(\%)$	T_2(K)	G_1/G_2	D_1/D_2	T_1/T_2
200	93.0	100.0	608.7	85.7	100.0	466.3	1.09	1.00	1.31
300	98.0	61.7	589.7	84.3	100.0	431.7	1.16	0.62	1.37
400	100.0	32.0	518.0	84.7	100.0	322.3	1.18	0.32	1.61

Table 3: Majority function: $n = 25, \kappa = 8, |\Xi| = 1000$

The results in the tables can be summed up as follows. First, for relative smaller sizes of the training sets initially given by the teacher (e.g. N = 300 in the case of n = 25), the incremental learning network achieved, on average, better generalization performance than the nonincremental one ($G_1/G_2 = 1.16$). In this case the same or smaller numbers of examples were used to train the network ($D_1/D_2 = 0.62$). Second, for larger sizes of initial training sets (e.g. for n = 25 and N = 400), the incremental learning network achieved better generalization than the nonincremental one did ($G_1/G_2 = 1.18$), using much smaller sizes of training sets than the original one ($D_1/D_2 = 0.32$). Third, the relative gain of the incremental learning network over the nonincremental one in D tends

set contained enough information. The basic idea was to guide the search process gradually in a global-to-local manner by first focusing on separating hyperplanes of the search space and then smoothing them. The focused search is performed by making use of the interestingness measure. In the selection phase, the interestingness provides a criterion for selecting critical patterns. In the adaptation phase, the interestingness provides a measure for giving priorities to novel examples so that they can be quickly adapted to the existing knowledge structure.

The significance of the focused incremental learning algorithm can be summarized as follows. First, the algorithm provides an automatic method for "improving the generalization by intelligently selecting the patterns in the training set" [1,4]. Second, the algorithm can be seen as an alternative to the *reversed learning* [8] for finding and building biases which reduces the number of potential solutions, without reducing the complexity of the problem. Third, the reduced training set can explain the implicit knowledge hidden in the connectionist network better than the original one, because the selected examples are representative of the problem. Fourth, the human teacher does not need to make an effort to carefully generate only good examples, because the learning algorithm itself can select useful examples. Furthermore the teacher himself has sometimes no idea of "good" examples. Finally, this means the human teaching of neural networks can be automated to some extent, provided that the learning algorithm possesses an additional capability of generating "plausible" (not absolutely "good") examples on the basis of existing knowledge and examples [10].

Acknowledgements

This work was supported in part by the POSCO Scholarship Society. We thank Jörg Kindermann, Heinz Mühlenbein, Knut Möller, Aleksander Linden, Sebastian Thrun, Thorsten Drabe and Paul Fimreite for their stimulating conversations and useful comments.

References

[1] Ahmad S & Tesauro G (1989) Scaling and generalization in neural networks: a case study, in: *Proc. of 1988 Connectionist Models Summer School*, pp. 3-10, Morgan Kaufmann.

[2] Chauvin Y (1989) A back-propagation algorithm with optimal use of hidden units, in: Touretzky DS (ed.) *Advances in Neural Information Processing*, pp. 519-526, Morgan Kaufmann.

[3] Hanson SJ & Pratt LY (1989) Comparing biases for minimal network construction with back-propagation, in: Touretzky DS (ed.) *Advances in Neural Information Processing*, pp. 177-185, Morgan Kaufmann.

[4] Huyser KA & Horowitz MA (1989) Generalization in connectionist networks that realize Boolean functions, in: *Proc. of 1988 Connectionist Models Summer School*, pp. 191-200, Morgan Kaufmann.

[5] Kruschke JK (1989) Creating local and distributed bottlenecks in hidden layers of back-propagation networks, in: *Proc. of 1988 Connectionist Models Summer School*, pp. 120-126, Morgan Kaufmann.

[6] le Cun Y (1989) *Generalization and Network Design Strategies*, CRG-TR-89-4, Dept. of Computer Science, University of Toronto.

[7] Moser MC & Smolensky P (1989) Skeletonization: A technique for trimming the fat from a network via relevance assessment, in: Touretzky DS (ed.) *Advances in Neural Information Processing*, pp. 107-115, Morgan Kaufmann.

[8] Pavel M, Gluck MA & Henkle V (1989) Constraints on adaptive networks for modeling human generalization, in: Touretzky DS (ed.) *Advances in Neural Information Processing*, pp. 2-10, Morgan Kaufmann.

[9] Rumelhart DE, Hinton GE & Williams RJ (1986) Learning internal representations by error propagation, in: Rumelhart DE & McClelland JL (eds.) *Parallel Distributed Processing*, Vol. I, pp. 318-362, MIT Press.

[10] Zhang BT (1990) *GENIAL: A Genetic Neural Evolutionary Model for Autonomous Learning and Self-Development*, Inst. for Computer Science, Div.I – Artificial Intelligence, University of Bonn, (in German).