Supervised Training Using an Unsupervised Approach to Active Learning

AP Engelbrecht and R Brits
Department of Computer Science, University of Pretoria, Pretoria, South Africa,
engel@driesie.cs.up.ac.za

Abstract. Active learning algorithms allow neural networks to dynamically take part in the selection of the most informative training patterns. This paper introduces a new approach to active learning, which combines an unsupervised clustering of training data with a pattern selection approach based on sensitivity analysis. Training data is clustered into groups of similar patterns based on Euclidean distance, and the most informative pattern from each cluster is selected for training using the sensitivity analysis incremental learning algorithm in (Engelbrecht and Cloete, 1999). Experimental results show that the clustering approach improves on standard active learning as presented in (Engelbrecht and Cloete, 1999).

Keywords: Active Learning, Incremental Learning, Pattern Informativeness, Sensitivity Analysis, Clustering

1. Introduction

Active learning algorithms have been developed as an approach to improve the performance of supervised neural networks (NN). Standard supervised NNs are passive learners which receive training patterns from a teacher, and train on all these patterns. In contrast to passive learning, active learning gives the learner the ability to select only the most informative patterns, and to train on these patterns.

Cohn, Atlas and Ladner define active learning as any form of learning in which the learning algorithm has some control over which part of the input space it receives information from (Cohn, Atlas and Ladner, 1994). An active learning strategy therefor allows the learner to dynamically select training examples, during training, from a candidate set as received from the teacher (or supervisor). The learner capitalizes on current attained knowledge to select examples from the candidate training set that are most likely to solve the problem, or that will lead to a maximum decrease in error. Rather than passively accepting training patterns from the teacher, the network is allowed to have some deterministic control over which examples to accept, and to guide the search for the most informative patterns. Figure 1 contrasts active learning and passive learning (also referred to as fixed set learning (FSL)).

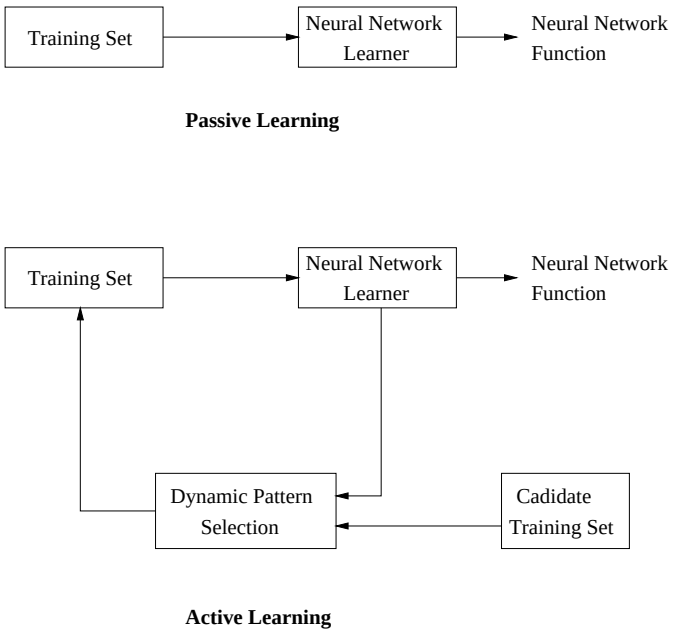


Figure 1. Active Learning vs Passive Learning

Provided that the added complexity of the pattern selection mechanism does not exceed the reduction in training computations (due to a reduction in the number of training patterns), training time will be reduced using active learning (Hunt and Deller, 1995; Sung and Niyogi, 1996; Zhang, 1994). Generalization can be improved provided that selected patterns contain enough information to learn the task (Cohn, 1994; Cohn, Atlas and Ladner, 1994; Seung, Opper and Sompolinsky, 1992; Sung and Niyogi, 1996; Zhang, 1994).

Two main approaches to active learning can be defined, namely *selective learning* (Engelbrecht and Cloete, 1998; Hampshire and Waibel, 1990; Hunt and Deller, 1995) and *incremental learning*. Selective learning selects a completely new training subset from the candidate training set at each subset selection interval, based on some measure of pattern informativeness. Each original candidate pattern is eligible for selection at each subset selection interval, regardless of whether the pattern has been selected at a previous subset selection interval. Incremental learning follows a similar approach, but with the exception that selected patterns are removed from the candidate training set, and added to the actual training set for the duration of training. The training set therefor grows during training, while the candidate training set shrinks.

This paper concentrates on incremental learning, and proposes an adaptation of an existing incremental learning algorithm, for regression

problems, to first cluster the candidate training set. The most informative pattern of each cluster is then selected and removed at each subset selection interval. The sensitivity analysis incremental learning algorithm (SAILA), introduced in (Engelbrecht and Cloete, 1999) is used for this purpose.

Several incremental learning algorithms have been developed, differing mainly in the measure of pattern informativeness. Most current incremental learning techniques have their roots in information theory, adapting Fedorov's optimal experiment design for NN learning (Sung and Niyogi, 1996; Cohn, 1994; MacKay, 1992; Fukumizu, 1996; Plutowski and White, 1993). The different information theoretic incremental learning algorithms are very similar, and differ only in whether they consider only bias, only variance, or both bias and variance terms to quantify pattern informativeness.

Cohn developed neural network Optimal Experiment Design (OED), where the objective is to select at each iteration a new pattern from a candidate set which minimizes the expectation of the mean squared error (MSE) (Cohn, 1994). This is achieved by minimizing output variance as estimated from the Fisher information matrix (Cohn, 1994; Cohn, Ghahramani and Jordan, 1996). The model assumes an unbiased estimator and considers only the minimization of variance. OED is computationally very expensive because it requires the calculation of the inverse of the information matrix.

MacKay proposed similar Information-Based Objective Functions for active learning, where the objective is to maximize the expected information gain by maximizing the change in Shannon entropy when new patterns are added to the actual training set, or by maximizing cross-entropy gain (MacKay, 1992; MacKay, 1992). Similar to OED, the maximization of information gain is achieved by selecting patterns that minimize the expected MSE. Information-Based Objective Functions also ignore bias, by minimizing only variance. The required inversion of the Hessian matrix makes this approach computationally expensive.

Plutowski and White proposed to select patterns that minimize the Integrated Squared Bias (ISB) (Plutowski and White, 1993). At each iteration, a new pattern is selected from a candidate set that maximizes the change, ΔISB, in the ISB. In effect, the patterns with error gradient most highly correlated with the error gradient of the entire set of patterns is selected. A noise-free environment is assumed and variance is ignored. Drawbacks of this method are the need to calculate the inverse of a Hessian matrix, and the assumption that the target function is known.

Sung and Niyogi proposed an information theoretic approach to active learning that considers both bias and variance (Sung and Niyogi,

1996). The learning goal is to minimize the expected misfit between the target function and the approximated function. The patterns that minimizes the expected squared difference between the target and approximated function are selected to be included in the actual training set. In effect, the net amount of information gained with each new pattern is then maximized. No assumption is made about the target function. This technique is computationally expensive, since it requires computations over two expectations, i.e. the a-posteriori distribution over function space, and the a-posteriori distribution over the space of targets one would expect given a candidate sample location.

A drawback of the incremental learning algorithms summarized above is that they rely on the inversion of an information matrix. Fukumizu showed that, in relation to pattern selection to minimize the expected MSE, the Fisher information matrix may be singular (Fukumizu, 1996). If the information matrix is singular, the inverse of that matrix may not exist. Fukumizu continues to show that the information matrix is singular if and only if the corresponding NN contains redundant units. Thus, the information matrix can be made non-singular by removing redundant hidden units. Fukumizu developed an algorithm that incorporates an architecture reduction algorithm with a pattern selection algorithm. This algorithm is complex due to the inversion of the information matrix at each selection interval, but ensures a non-singular information matrix.

Approximations to the information theoretical incremental learning algorithms can be used. Zhang illustrates that information gain is maximized when a pattern is selected whose addition leads to the greatest decrease in MSE (Zhang, 1994). Zhang developed Selective Incremental Learning where training starts on an initial subset which is increased during training by adding additional subsets, where each subset contains those patterns with largest errors. Selective Incremental Learning has a very low computational overhead, but is negatively influenced by outlier patterns since these patterns have large errors.

Dynamic Pattern Selection, developed by Röbel (Röbel, 1994; Röbel, 1994), is very similar to Zhang's Selective Incremental Learning. Röbel defines a generalization factor on the current training subset, expressed as $\mathcal{E}_G/\mathcal{E}_T$ where $\mathcal{E}_G$ and $\mathcal{E}_T$ are the MSE of the test set and the training set respectively. As soon as the generalization factor exceeds a certain threshold, patterns with highest errors are selected from the candidate set and added to the actual training set. Testing against the generalization factor prevents overfitting of the training subset. A low overhead is involved.

The incremental learning algorithms above can be applied to both regression and classification problems. Incremental learning algorithms

have also been developed for classification problems specifically, where decision boundaries are utilized to guide the search for optimal training subsets. Cohn, Atlas and Ladner developed Selective Sampling, where patterns are sampled only within a *region of uncertainty* (Cohn, Atlas and Ladner, 1994). Cohn *et al* proposed an SG-network (most specific / most general network) as an approach to compute the region of uncertainty. Two separate networks are trained: one to learn a "most specific" concept s consistent with the given training data, and the other to learn a "most general" concept, g. The region of uncertainty is then all patterns p such that $s(p) \neq g(p)$. In other words, the region of uncertainty encapsulates all those patterns for which s and g present a different classification. A new training pattern is selected from this region of uncertainty and added to the training set. After training on the new training set, the region of uncertainty is recalculated, and another pattern is sampled according to some distribution defined over the uncertainty region - a very expensive approach. To reduce complexity, the algorithm is changed to select patterns in batches, rather than individually. An initial pattern subset is drawn, the network is trained on this subset, and a new region of uncertainty is calculated. Then, a new distribution is defined over the region of uncertainty that is zero outside this region. A next subset is drawn according to the new distribution and added to the training set. The process repeats until convergence is reached.

Query-Based Learning, developed by Hwang, Choi, Oh and Marks (Hwang, Choi, Oh and Marks, 1991) differs from Selective Sampling in that Query-Based Learning generates new training data, for classification problems, in the region of uncertainty. The objective is to increase the steepness of the boundary between two distinct classes by narrowing the regions of ambiguity. This is accomplished by inverting the NN output function to compute decision boundaries. New data in the vicinity of boundaries are then generated and added to the training set.

Seung, Opper and Sompolinsky proposed Query by Committee, for classification problems (Seung, Opper and Sompolinsky, 1992). The optimal training set is built by selecting one pattern at a time from a candidate set based on the principle of maximal disagreement among a committee of learners. Patterns classified correctly by half of the committee, but incorrectly by the other half, are included in the actual training set. Query by Committee is time consuming due to the simultaneous training of several networks, but will be most effective for ensemble networks.

The remainder of this paper is organized as follows: Section 2 provides an overview of the sensitivity analysis incremental learning al-

gorithm introduced in (Engelbrecht and Cloete, 1999). The new algorithm, cluster-SAILA (CSAILA) is presented in section 3. A comparison of CSAILA, SAILA and FSL is given in section 4.

2. Sensitivity Analysis Incremental Learning

The sensitivity analysis incremental learning algorithm in (Engelbrecht and Cloete, 1999) were developed for regression problems. SAILA, can, however, also be applied to classification problems. For the purposes of this paper, the focus remains on regression problems for comparison with the regression problems used in (Engelbrecht and Cloete, 1999).

The sensitivity analysis incremental learning algorithm defines pattern informativeness as the sensitivity of the NN output to perturbations in the input values of that pattern (Engelbrecht and Cloete, 1999). That is,

$$\Phi^{(p)} = ||\vec{S}_o^{(p)}||_\infty = \max_{k=1,\cdots,K}\{|S_{o,k}^{(p)}|\} \tag{1}$$

where $\Phi^{(p)}$ is the informativeness of pattern p, $\vec{S}_o^{(p)}$ is the output sensitivity vector for pattern p, and $S_{o,k}^{(p)}$ refers to the sensitivity of a single output unit o_k to changes in the input vector $\vec{z}$; K is the total number of output units.

The output sensitivity vector is defined as

$$\vec{S}_o^{(p)} = ||S_{oz}^{(p)}||_2 \tag{2}$$

where $S_{oz}^{(p)}$ is the output-input layer sensitivity matrix. Each element $S_{oz,ki}^{(p)}$ of the sensitivity matrix is defined as (assuming differentiable activation functions)

$$S_{oz,ki}^{(p)} = \frac{\partial o_k}{\partial z_i^{(p)}} \tag{3}$$

Each element k of $\vec{S}_o^{(p)}$ is then computed as

$$S_{o,k}^{(p)} = \sqrt{\sum_{i=1}^{I}(S_{oz,ki}^{(p)})^2} \tag{4}$$

At each subset selection interval, SAILA selects only the most informative pattern p, i.e.

$$p = \{p \in D_C | \Phi^{(p)} = \max_{q=1,\cdots,P_C}\{\Phi^{(q)}\}\} \tag{5}$$

where D_C is the current candidate training set and P_C is the number of patterns remaining in D_C. The pattern p is then removed from D_C and added to the current training subset D_T. Training continues on the training subset D_T until any one of the following subset selection criteria becomes true, upon which a new informative pattern is selected:

- The maximum number of training epochs on the current training subset has been exceeded.
- A sufficient decrease in training error has been achieved.
- The average decrease in error per epoch is too small.
- Overfitting of the current training subset has been detected.

3. Cluster-SAILA

The cluster sensitivity analysis incremental learning algorithm (CSAILA) is an extension of SAILA overviewed in the previous section. SAILA is extended to first cluster the patterns in the candidate training set. After the clustering process, SAILA is applied to each of the clusters to find and remove the most informative pattern of each cluster. If the candidate training set is divided into C clusters, C informative patterns are selected and added to the training subset. Each selected pattern is removed from the corresponding cluster. When only one cluster is used, CSAILA and SAILA are equivalent.

The success of CSAILA lies in the number of clusters used. The cluster algorithm implemented for CSAILA dynamically grows the number of clusters based on the variance in Euclidean distance of all the patterns grouped in a cluster from the cluster center. If the variance in Euclidean distance exceeds a user specified threshold, θ, a new cluster is added. The complete cluster algorithm is given below:

1. Initialization:
   a) Find the minimum z_i^{min} and maximum z_i^{max} values of each input attribute z_i over the candidate training set D_C.
   b) Initialize the cluster threshold θ and the initial number of clusters C.
   c) Initialize the C cluster reference vectors $\vec{\rho}_1, \cdots, \vec{\rho}_C$. Each reference vector is initialized randomly such that each element $\rho_{c,i}$ of reference vector $\vec{\rho}_c$ falls within the minimum and maximum values of the corresponding input attributes. That is,

$$\rho_{c,i} \sim U(z_i^{min}, z_i^{max}) \tag{6}$$

2. For each pattern $p \in D_C$

   a) Calculate the Euclidean distance

$$\epsilon_c^{(p)} = \sqrt{\sum_{i=1}^{I} (z_i^{(p)} - \rho_{c,i})^2} \tag{7}$$

   to each cluster reference vector $\vec{\rho}_c$.

   b) Find the closest cluster $\rho_c = \min_{c=1,\cdots,C}\{\epsilon_c^{(p)}\}$ and add pattern p to cluster ρ_c

   c) Adjust the reference vector $\vec{\rho}_c$ of cluster ρ_c:

$$\rho_{c,i}(t) = \rho_{c,i}(t-1) + \eta(z_i^{(p)} - \rho_{c,i}(t-1)) \;\; \forall i = 1, \cdots, I \tag{8}$$

   where η is the learning rate.

3. Test for convergence

   a) Calculate the variance $\sigma^2_{\epsilon_c}$ in Euclidean distance of each $p \in \rho_c$ for each cluster.

   b) If $\sigma^2_{\epsilon_c} \leq \theta$ for all $c = 1, \cdots, C$, the clustering algorithm terminates.

   c) If $\sigma^2_{\epsilon_c} > \theta$ for one of the clusters, then add an additional cluster with a randomly generated reference vector, and goto step 2.

After execution of the clustering algorithm, the incremental learning algorithm uses the SAILA approach to select at each subset selection interval the most informative pattern from each cluster.

The efficiency of CSAILA depends directly on the number of clusters, as controlled by the cluster variance threshold θ. The larger the value of θ, the less clusters are generated, while more clusters are formed for smaller θ values. More clusters mean that more patterns will be selected for training, thereby increasing the total number of pattern presentations and increasing the computational complexity of training. The best value of θ should be obtained through cross-validation.

The next section presents results to compare the new CSAILA algorithm with SAILA and FSL.

4. Experimental Results

This section presents results of CSAILA in comparison with SAILA and FSL, using gradient descent for training purposes. The performance of

the different algorithms are compared based on generalization performance, convergence and complexity, on a set of regression problems. For this purpose, the following functions have been used (as illustrated in figure 2):

- Function F1:
$$F1: \quad f(z) = z^2 \tag{9}$$
where $z \sim U(-1, 1)$. All the output values are in the range $[0, 1]$.

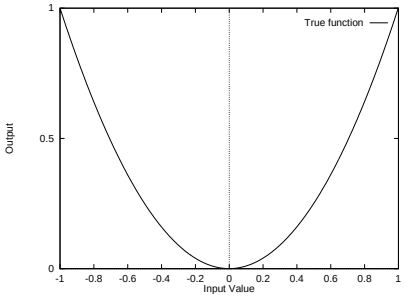
- Function F2:
$$F2: \quad f(z) = \sin(2\pi z)e^{-z} + \zeta \tag{10}$$
where $z \in U(-1, 1)$, and $\zeta \sim N(0, 0.1)$. Output values were scaled to the range $[0, 1]$.

- Function F3:
$$F3: \quad f(z_1, z_2) = \frac{1}{2}(z_1^2 + z_2^2) \tag{11}$$
where $z_1, z_2 \sim U(-1, 1)$, and all outputs are in the range $[0, 1]$.

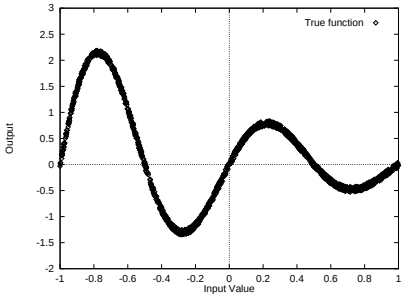
- Time series TS1 (the Henon-map):
$$\begin{aligned} TS1: o_t &= z_t \\ z_t &= 1 + 0.3z_{t-2} - 1.4z_{t-1}^2 \end{aligned} \tag{12}$$
where $z_1, z_2 \sim U(-1, 1)$. The output values o_t were scaled such that $o_t \in [0, 1]$.

- Time series TS2:
$$\begin{aligned} TS2: o_t &= z_t \\ z_t &= 0.3z_{t-6} - 0.6z_{t-4} + 0.5z_{t-1} + 0.3z_{t-6}^2 - 0.2z_{t-4}^2 + \zeta_t \end{aligned} \tag{13}$$
where $z_t \sim U(-1, 1)$ for $t = 1, \cdots, 10$, and $\zeta_t \sim N(0, 0.05)$ is zero-mean noise sampled from a normal distribution. All output values were scaled to the range $[0, 1]$.

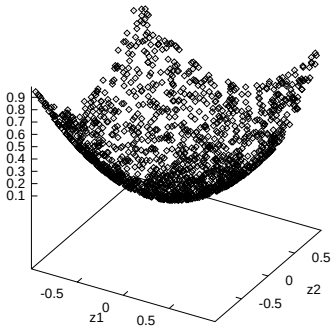
Table I presents a summary of all these functions, together with the NN architecture and parameters used. For each of these functions, 50 simulations were executed. Each simulation consists of a CSAILA, SAILA and FSL run, using the same data sets, initial weights, architecture, learning rate and momentum (refer to table I). The candidate training set D_C, test set D_G and validation set D_V were randomly created to be independent data sets.



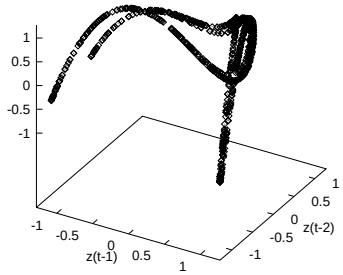
a. Function F1



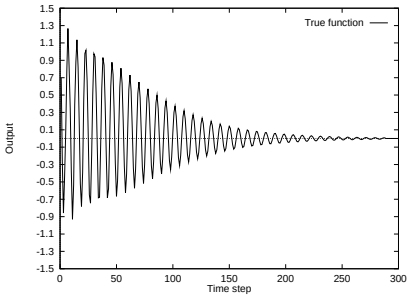
b. Function F2



c. Function F3



d. Time Series TS1



e. Time Series TS2

Figure 2. Functions used to compare CSAILA, SAILA and FSL

Table I. Summary of Test Functions and Network Parameters

Problem	**F1**	**F2**	**F3**	**TS1**	**TS2**
Training Set	120	600	600	600	180
Test Set	40	200	200	200	60
Validation Set	40	200	200	200	60
Learning Rate	0.1	0.1	0.1	0.05	0.05
Momentum	0.9	0.9	0.9	0.9	0.9
Maximum Epochs	2000	2000	2000	4000	1000
Architecture	1-3-1	1-10-1	2-5-1	2-5-1	10-10-1
Cluster Variance Threshold	0.01	0.01	0.05	0.01	0.01
Initial Number of Clusters	3	3	3	3	3

Generalization Performance

The training error and generalization performance of the training algorithms are compared in table II. The second and third columns respectively shows the average training error $\overline{E}_T$ and average generalization error $\overline{E}_G$ over the 50 simulations (these errors are reported as the MSE over the respective data sets), together with a 95% confidence interval. Both SAILA and CSAILA consistently showed better performance than FSL. Furthermore, CSAILA performed better than SAILA for functions F1, F2 and F3 (with substantial improvements for the latter two functions). For TS1, CSAILA and SAILA achieved the same training accuracy, but a much better generalization was achieved by CSAILA. For TS2, FSL and CSAILA had approximately the same training error, but FSL substantially overfitted with a bad generalization. SAILA performed the best for TS2. However, even though CSAILA reached a higher training error than SAILA, a generalization performance comparable to that of SAILA has been reached.

The improvements in performance by CSAILA are attributed to the larger set of informative patterns selected for training. The most informative patterns as used by CSAILA span a wider area of input space compared to that of SAILA.

Table II. Error Performance Measures

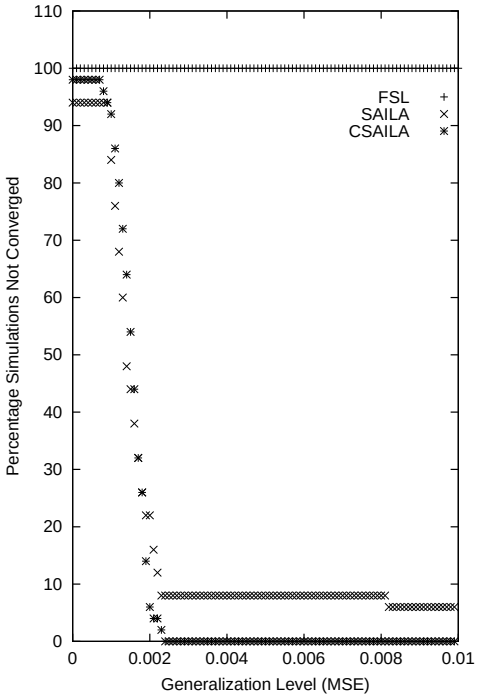
Problem		$\overline{E}_T$	$\overline{E}_G$	E_G^{best}
F1:	*FSL*	0.160 ± 0.040	0.158 ± 0.040	0.158
	SAILA	0.0017 ± 0.0008	0.0022 ± 0.0053	0.0009
	CSAILA	0.0015 ± 0.00022	0.0015 ± 0.00035	0.00075
F2:	*FSL*	0.027 ± 0.003	0.027 ± 0.003	0.030
	SAILA	0.013 ± 0.019	0.012 ± 0.015	0.001
	CSAILA	0.0053 ± 0.0065	0.0062 ± 0.0043	0.0012
F3:	*FSL*	0.055 ± 0.023	$0.056 \pm 0,024$	0.057
	SAILA	0.0034 ± 0.01	0.0014 ± 0.002	0.00036
	CSAILA	0.00073 ± 0.0004	0.00073 ± 0.0004	0.00038
TS1:	*FSL*	0.083 ± 0.034	0.084 ± 0.034	0.085
	SAILA	0.00061 ± 0.0002	0.00076 ± 0.0004	0.00019
	CSAILA	0.00068 ± 0.0002	0.00053 ± 0.0002	0.00026
TS2:	*FSL*	0.009 ± 0.001	0.027 ± 0.004	0.027
	SAILA	0.0047 ± 0.009	0.0045 ± 0.012	0.00026
	CSAILA	0.0087 ± 0.011	0.005 ± 0.0079	0.00029

Convergence

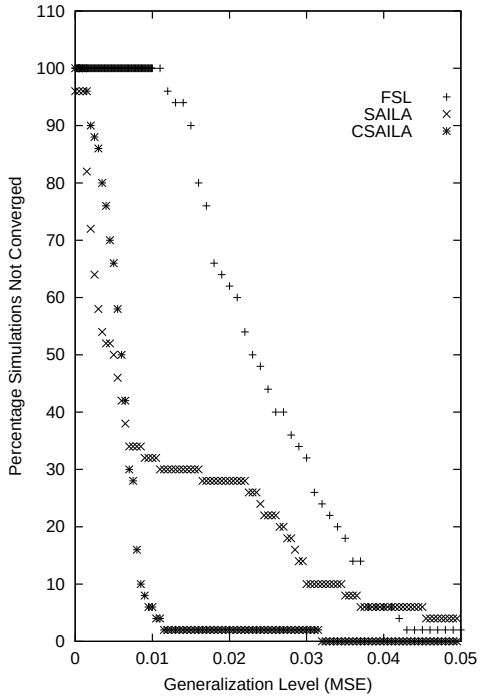
The convergence characteristics of CSAILA, SAILA and FSL are compared in figure 3 for the function approximation problems and in figure 4 for the time series problems. These figures plot the percentage of simulations that did not reach specific generalization errors. For all the problems studied in this paper, CSAILA and SAILA exhibited consistently better convergence characteristics than FSL. For all the functions, except for TS2, CSAILA had more converged simulations than SAILA. It is for low generalization levels of less than 0.004 that CSAILA improved on SAILA for TS2.

Complexity

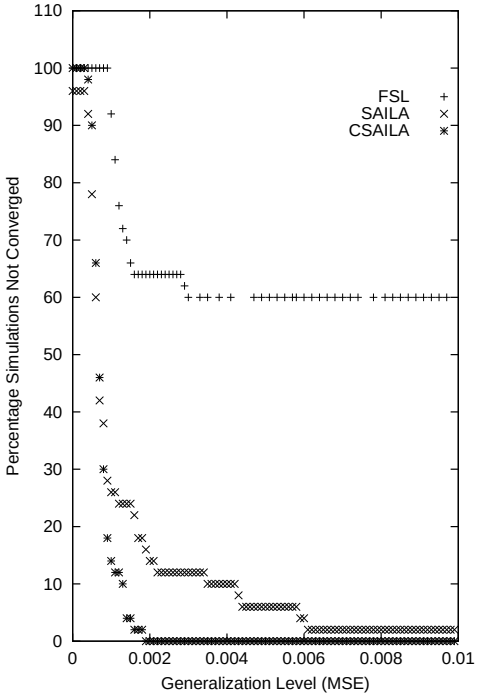
The complexity of CSAILA and SAILA is directly related to the number of patterns selected from the candidate training set. The less patterns selected, the less weight updates need to be made. Even with the little added complexity of the sensitivity analysis pattern selection approach (all information used by the selection approach is already



a. Function F1

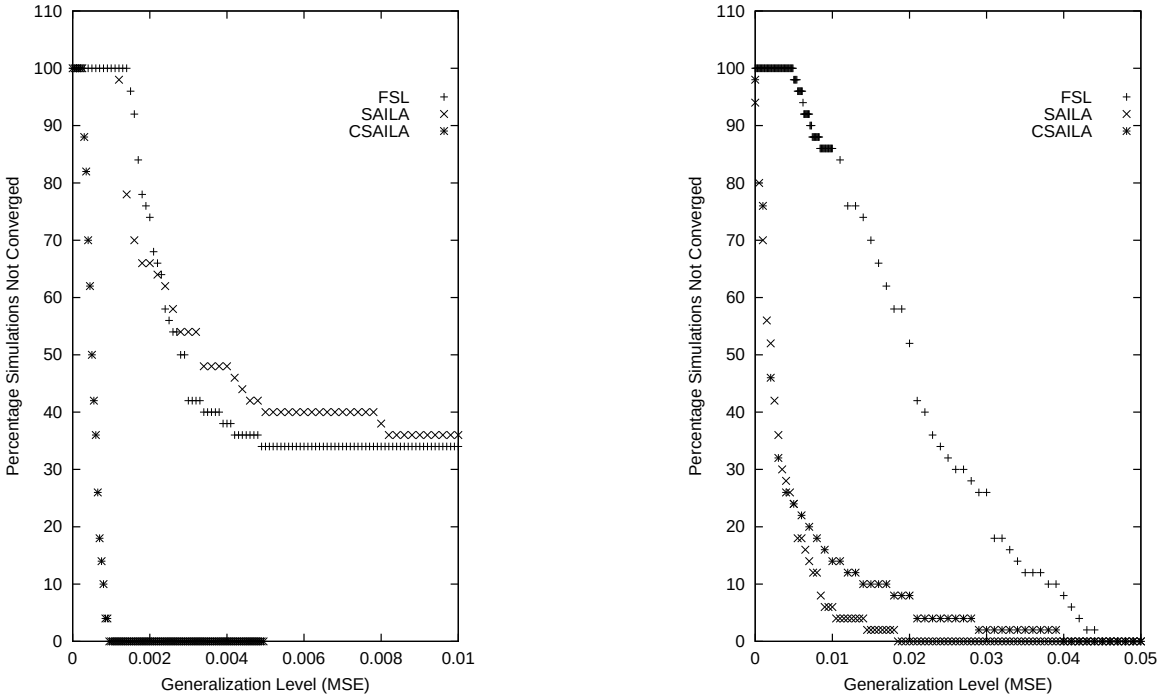


b. Function F2



c. Function F3

Figure 3. Percentage simulations that did not converge to generalization levels for function approximation problems



a. Time Series TS1 b. Time Series TS2

Figure 4. Percentage simulations that did not converge to generalization levels for time series problems

available from the training equations), substantial reductions in training set sizes resulted in large reductions in computational complexity. Table III summarizes for selected epochs the percentage of the original candidate set that was used by CSAILA and SAILA for training, in comparison with the original size of the candidate training set (as given in the last column). Table III shows that both CSAILA and SAILA used substantially less patterns, therefor less pattern presentations. The second last column lists the percentage reduction in the total number of pattern presentations (i.e. the total number of backward propagations to update weights) at the end of training, with reference to the total number of pattern presentations for FSL. Both CSAILA and SAILA resulted in large reductions in the total number of pattern presentations. With the exception of function F2, the two incremental learning algorithms had approximately the same reductions. For F2, SAILA used much less patterns than CSAILA, but with the disadvantage of having a worse generalization performance (refer to table II).

Table III. Summary of Training Set Sizes and Total Number of Pattern Presentations

Problem		D_T After Epoch 50	200	600	1000	2000	Percentage Reduction	P_C
F1:								
	SAILA	3.0%	14.2%	48.3 %	53.2%	83.3%	36.9%	120
	CSAILA	5.8%	5.8%	39.2%	56.7%	95.8%	32.3%	
F2:								
	SAILA	1.1%	2.5%	7.5%	12.8%	23.8%	83.8%	600
	CSAILA	2.0%	8.5%	16.0%	21.0%	66.5%	58.7%	
F3:								
	SAILA	1.6%	5.0%	19.3%	34.5%	71.0%	53.3%	600
	CSAILA	6.7%	10.8%	21.2%	27.7%	70.8%	53.9%	
TS1:								
	SAILA	1.3%	7.3%	26.7%	45.0%	85.5%	57.3%	600
	CSAILA	7.0%	7.0%	16.7%	39.9%	69.0%	53.5%	
TS2:								
	SAILA	5.0%	16.1%	22.5%	97.2%	-	49.5%	180
	CSAILA	7.8%	25.0%	56.7%	80.0%	-	41.2%	

5. Conclusions

This paper presented a new approach to active learning, which combines unsupervised clustering of the candidate training set with a previously developed sensitivity analysis-based incremental learning algorithm. Clustering of the original training data, and selection of the most informative patterns from each of the clusters improved the performance of the incremental learning algorithm. The clustering approach to incremental learning (CSAILA) consistently showed better convergence results than the original incremental learning algorithm (SAILA) and fixed set learning (FSL). CSAILA resulted in better generalization performance that SAILA ad FSL.

Further research is necessary in determining the influence of the number of clusters on performance. More clusters provide more training information per training subset selection interval, but also increases computational complexity since more pattern presentations are made during training. However, too many clusters approximates the perfor-

mance of FSL, while too few clusters approximates the performance of SAILA.

References

Cohn, D. A. Neural Network Exploration using Optimal Experiment Design. AI Memo No 1491, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1994.

Cohn, D. A., L. Atlas and R. Ladner. Improving Generalization with Active Learning. *Machine Learning*, 15:201-221, 1994.

Cohn, D. A., Z. Ghahramani and M. I. Jordan. Active Learning with Statistical Models. *Journal of Artificial Intelligence Research*, 4:129-145, 1996.

Engelbrecht, A. P. and I. Cloete. Selective Learning using Sensitivity Analysis. In *Proceedings of IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, 1150-1155, 1998.

Engelbrecht, A. P. and I. Cloete Incremental Learning using Sensitivity Analysis In *IEEE International Joint Conference on Neural Networks*, Washington DC, USA, paper 380, 1999.

Fukumizu, K. Active Learning in Multilayer Perceptrons. In D. S. Touretzky, M. C. Mozer, M .E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, 8:295-301, 1996.

Hampshire, J. B. and A. H. Waibel. A Novel Objective Function for Improved Phoneme Recognition using Time-Delay Neural Networks. *IEEE Transactions on Neural Networks*, 1(2):216-228, 1990.

Hunt, S. D. and J. R. Deller (jr). Selective Training of Feedforward Artificial Neural Networks using Matrix Perturbation Theory. *Neural Networks*, 8(6):931-944, 1995.

Hwang, J-N., J. J. Choi, S. Oh and R. J. Marks II. Query-Based Learning Applied to Partially Trained Multilayer Perceptrons. *IEEE Transactions on Neural Networks*, 2(1):131-136, 1991.

MacKay, D. J. C. *Bayesian Methods for Adaptive Models.* PhD Thesis, California Institute of Technology, 1992.

MacKay, D. J. C. Information-Based Objective Functions for Active Data Selection. *Neural Computation*, 4:590-604, 1992.

Plutowski, H. and H. White. Selecting Concise Training Sets from Clean Data. *IEEE Transactions on Neural Networks*, 4(2):305-318, 1993.

Röbel, A. Dynamic Pattern Selection: Effectively Training Backpropagation Neural Networks. *International Conference on Artificial Neural Networks*, 1:643-646, 1994.

Röbel, A. Dynamic Pattern Selection for Faster Learning and Controlled Generalization of Neural Networks. European Symposium on Artificial Neural Networks, 1994.

Seung, H. S., M. Opper and H. Sompolinsky. Query by Committee. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, 287-299, 1992.

Sung, K. K. and P. Niyogi. A Formulation for Active Learning with Applications to Object Detection. AI Memo No 1438, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1996.

Zhang, B-T. Accelerated Learning by Active Example Selection. *International Journal of Neural Systems*, 5(1):67-75, 1994.