Selecting Inputs for Modeling Using Normalized Higher Order Statistics and Independent Component Analysis

Andrew D. Back and Thomas P. Trappenberg

***Abstract*—The problem of input variable selection is well known in the task of modeling real-world data. In this paper, we propose a novel model-free algorithm for input variable selection using independent component analysis and higher order cross statistics. Experimental results are given which indicate that the method is capable of giving reliable performance and that it outperforms other approaches when the inputs are dependent.**

***Index Terms*—Higher order statistics, independent component analysis, input variable selection.**

I. Introduction

In many real-world modeling problems, for example in the context of biomedical, industrial, or environmental systems, a problem can occur when developing multivariate models and the best set of inputs to use are not known. This is particularly true when using neural networks. In this case, unrequired inputs can significantly increase learning complexity. Input variable selection (IVS) is aimed at determining which input variables are required for a model. The task is to determine a set of inputs which will lead to an optimal model in some sense. Problems which can occur due to poor selection of inputs include the following.

- As the input dimensionality increases, the computational complexity and memory requirements of the model increase.
- Learning is more difficult with unrequired inputs.
- Misconvergence and poor model accuracy may result from additional unrequired inputs.
- Understanding complex models is more difficult than simple models which give comparable results.

Methods of input variable selection can be categorized into *model-based* and *model-free* methods [1]. *Model-based* methods typically involve selecting a model, choosing the inputs to use, optimizing the parameters, and then measuring some cost function. The inputs are changed and then the procedure is repeated. A test is used to choose which inputs to use based on these results. Model-based input variable selection schemes are often linked to the idea of *pruning* networks [2]–[11]. Neural-network pruning methods based on principal component analysis (PCA) have also been proposed. Leen *et al*. [15], [16] investigated the idea of using PCA to remove variables which have the lowest variance.

Model-free methods are based on performing a statistical test between the subsets of input variable(s) and the desired output(s) from the model. A very good example of a model-free IVS method based on *mutual information* is given in [1]. In contrast to other model-based methods, the idea in this case was to develop a framework for selecting inputs which was not based on a particular model. Relevant inputs are found by estimating the mutual information between the inputs and the desired outputs. This approach requires the numerical estimation of the joint and marginal densities. A measure of mutual information is obtained by calculating the Kullback–Leibler distance from the estimated densities. The work we present in this paper follows along similar lines and we also present a model-free method.

We follow the convention adopted previously, of using the term "model-free," even though for the method we present here, there are a few parameters left to estimate. The terminology comes from the idea that model-based methods implement a full prediction/classification model in order to select the input variables, while model-free methods refer to techniques that have either no parameters or only a small number in comparison to the intended model [1]. Moreover, model-free methods can often be viewed as having the express purpose of selecting the input variables (or groups of variables) to which some observed output variable is causally related. Model-based methods are, on the other hand, methods which select inputs based on how well they can lead to some model meeting a performance criterion. These differences have lead to the commonly used convention of terminologies which we continue to use here.

In this paper we address the issue of possible dependence between observed input variables. Dependent inputs usually leads to an overestimation of the number of inputs required, which, for neural network models is not desirable. We propose the use of independent component analysis (ICA) as a technique for deriving effective model-free input variable selection algorithms. In order to assess the dependence between inputs and the desired system output, we use a method based on higher order cross correlations, normalized in such a manner as to allow their direct comparison.

This paper is organized as follows. In Section II, we propose an input variable selection algorithm. In Section III, we give simulation examples which indicate the peformance of the algorithm. Conclusions are given in Section IV.

Manuscript received September 14, 1999; revised May 16, 2000 and December 13, 2000.

A. D. Back and T. P. Trappenberg were with the RIKEN Brain Science Institute, Saitama 351-0198, Japan (e-mail: andrew.back@usa.net).

T. P. Trappenberg is now the Department of Experimental Psychology, University of Oxford, Oxford OX1 3UD, U.K. (e-mail: Thomas.Trappenberg@psy.ox.ac.uk).

Publisher Item Identifier S 1045-9227(01)02046-X.

II. ICAIVS—An ICA Input Variable Selection Algorithm

A. Assumptions

The usual IVS problem can be described mathematically as follows. A system F_o receives input from the variables $\mathbf{z}(t) = [z_1(t) \cdots z_p(t)]'$ and produces an output $y(t)$

$$y(t) = F_o(\mathbf{z}(t)). \tag{1}$$

It is assumed that the system F_o can be approximated arbitrarily well by a linear or nonlinear functional map. To estimate F_o, measurements $\mathbf{x}(t) = [x_1(t) \cdots x_n(t)]'$ are taken with the assumption that

$$\mathbf{z}(t) \subseteq \mathbf{x}(t). \tag{2}$$

The usual model building approach is to apply an input variable selection procedure to obtain a set of model inputs $\mathbf{x}_a(t) \subseteq \mathbf{x}(t)$ where ideally $\mathbf{x}_a(t) = \mathbf{z}(t)$. Hence a model can be written as

$$\hat{y}(t) = F(\mathbf{x}_a(t); \boldsymbol{\theta}) \tag{3}$$

where F is a functional map parametrized by $\boldsymbol{\theta}$.

However, it is clear that the above assumptions may not be valid in practice. It is likely that the measurements $\mathbf{x}(t)$ are not a strict superset of actual inputs $\mathbf{z}(t)$ to the system. It seems likely that we would more often observe data which has been *filtered* in some manner, relative to the true inputs. For example, suppose there exists $\mathbf{z}_L(t) : \mathbf{z}(t) \subseteq \mathbf{z}_L(t)$, then the observed data may be written as

$$\mathbf{x} = \mathbf{A}(z)\mathbf{z}_L \tag{4}$$

where $\mathbf{A}(z)$ is a multivariate filter. A simpler variation of this is where no temporal filtering of the signals is involved, giving

$$\mathbf{x} = \mathbf{A}\mathbf{z}_L \tag{5}$$

where in this case $\mathbf{A}$ is a mixture matrix containing scalar terms only.

In this situation, it is important to recognize that there does not exist any subset $\mathbf{x}_a(t) \subseteq \mathbf{x}(t)$ such that $\mathbf{x}_a(t) = \mathbf{z}(t)$. Moreover, due to the components of $\mathbf{z}(t)$ appearing in $\mathbf{x}$ due to $\mathbf{A}$, there will tend to be an overestimation of the number of inputs required.

To avoid this overestimation problem the observed signals need to be inverse filtered or demixed, so as to obtain an estimate of the postulated $\mathbf{z}_L$. This will then allow an input variable selection method to be applied with less chance of overestimating the number of signals. Thus we have

$$\mathbf{z}_L = \mathbf{W}\mathbf{x} \tag{6}$$

$$\mathbf{z} = \mathbf{G}\mathbf{z}_L \tag{7}$$

where $\mathbf{W}$ is an estimate of $\mathbf{A}^{-1}$ and $\mathbf{G}$ is a sparse matrix which selects the desired subset of inputs to the model according to a particular algorithm.

However, while we require $\mathbf{W}$, we may have no explicit knowledge of $\mathbf{A}$. Recently, this seemingly difficult problem of estimating $\mathbf{A}$ and hence $\mathbf{z}_L$ from only the observed output measurements $\mathbf{x}$ has been solved. A family of mathematical techniques known as ICA or blind source separation has been shown to give exactly the solution we require [12], [13]. Based on the assumption of either temporal or spatial independence of the channels, ICA estimates a demixing matrix and the input signals. One ICA method is to estimate the mutual information between the signals and adjust the estimated matrix $\mathbf{W}$ to give outputs which are maximally independent. We consider this approach below as a means of providing an improved method of input variable selection.

B. Determining Statistical Dependence

We propose to use ICA to make the input variables as mutually independent as possible. Moreover, using ICA allows us to derive model-free IVS algorithms based on statistical dependence tests. The basic strategy we suggest is to apply ICA to estimate the independent inputs $\mathbf{z}$ from $\mathbf{x}$ and then derive a statistical test to determine the desired subset of input variables $\mathbf{z}_a$.

One approach for determining statistical dependence is to estimate the mutual information between two signals x and y given by

$$I(x;y) = \int p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy \tag{8}$$

This is also known as the cross-entropy or the Kullback–Leibler divergence between the joint probability density function (pdf) of (x,y) given by $p(x,y)$ and the product of the marginal pdfs $p(x)$, $p(y)$. This may be implemented by estimating the pdfs in terms of the cumulants of the signals, for example, using the Gram-Charlier expansion.

Estimating mutual information is difficult however, due to the large amount of data that may be required and not knowing the order of cumulants to use. Fourth-order cumulants are commonly used, but this may not be sufficient to form an accurate approximation. Indeed, to approximate a pdf far from gaussian, it is probable that a very large number of terms will be required [14].

Since we only require a relatively simple binary decision to be made about the dependence or otherwise of signals, it is not necessary to compute a precise value for the mutual information. Instead, the higher order cross cumulants[1] of multiple variables can be used directly up to some suitable order, to determine the statistical dependence of the signals.

C. Normalized Higher Order Cross Correlations

We propose to determine the input variables required for a given modeling problem using the simple approach of:

1) making the inputs as independent as possible;
2) testing all possible[2] input combinations to find the required subset.

The method used to find the required inputs uses higher order cross cumulants, up to a specified order among the individual

[1]In contrast to the often quoted first-order cross cumulant measure $c_{x_1 \cdots x_n}(\tau_1, \ldots, \tau_n)$, of n variables or the mth-order cumulant of a single variable. Since we are seeking to determine the statistical dependence between variables not just the correlation between variables, it is necessary to use higher order cross cumulants.

[2]Or as many input combinations as required.

terms. This statistical measure can be used to establish the independence or otherwise of non-Gaussian signals. These cumulants are defined as

$$\begin{aligned}
\mathbf{C}_{xy}(k) =&[c_{x_1y}(k),\ldots,c_{x_ny}(k)]^{\mathrm{T}}\\
&\vdots\\
\mathbf{C}_{x^py}(k) =&[c_{x_1^py}(k),\ldots,c_{x_n^py}(k)]^{\mathrm{T}}\\
\mathbf{C}_{x_1x_2y}(k) =&[c_{x_1x_2y}(k),c_{x_1x_3y}(k),\ldots,c_{x_1x_ny}(k)\\
&\quad c_{x_2x_3y}(k),\ldots,c_{x_{n-1}x_ny}(k)]^{\mathrm{T}}\\
&\vdots\\
\mathbf{C}_{x_1^px_2^py}(k) =&[c_{x_1^px_2^py}(k),c_{x_1^px_3^py}(k),\ldots,c_{x_1^px_n^py}(k)\\
&\quad c_{x_2^px_3^py}(k),\ldots,c_{x_{n-1}^px_n^py}(k)]^{\mathrm{T}}\\
&\vdots\\
\mathbf{C}_{x_1x_2x_3y}(k) =&[c_{x_1x_2x_3y}(k),c_{x_1x_2x_4y}(k),\ldots,c_{x_1x_2x_ny}(k)\\
&\quad c_{x_1x_3x_4y}(k)\ldots,c_{x_{n-2}x_{n-1}x_ny}(k)]^{\mathrm{T}}\\
&\vdots\\
\mathbf{C}_{x_1^px_2^px_3^py}(k) =&[c_{x_1^px_2^px_3^py}(k),c_{x_1^px_2^px_4^py}(k),\ldots,c_{x_1^px_2^px_n^py}(k)\\
&\quad c_{x_1^px_3^px_4^py}(k)\ldots,c_{x_{n-2}^px_{n-1}^px_n^py}(k)]^{\mathrm{T}}
\end{aligned}$$

where the cross cumulant vectors are between the inputs $x_1, x_2, \ldots, x_n$ at time $t-k$ and the output y. For convenience, we will use the notation $C_m(k) = C_{x_1\ldots x_my}(k)$. There are two broad cases that can be considered at this point:

1) models with only instantaneous inputs. In this case we use $\mathbf{C}_m(0)$;
2) models with delayed inputs. Here we test input variable $\{x_i(t-k)\}\, k = 0,\ldots,p, i = 1,\ldots,n$ against the system output $y(t)$, by examining the points in the cross-cumulant space given by elements of the vector $\mathbf{C}_m(k)$.

Our aim is to combine various cumulant measurements and thereby apply a decision function to determine the relative dependence of each input subset on the model output(s). Note, however, that if we wish to combine the cumulant measures, it is necessary to normalize them first. The reason for this is that the numerical scale between cumulants of different order are unrelated. Only the relative values between the cumulants within the same order are essential to determine their importance.

Hence we normalize[3] each of the cumulants in order to compare and combine them in a reasonable manner. The following normalization steps are applied to the cumulants:

1) Zero mean signals: $x_i = x_i - E[x_i]$.
2) For cumulants using second and higher even cross-order terms, normalize as: $x_i^{2n} = x_i^{2n} - E[x_i^{2n}] \cdots n = 1, 2, \ldots$
3) Sign of data: $x_i = \mathrm{sgn}(x_i)$
4) Renormalize individual cumulants. This step is necessary in order to compare the cumulants with each other. In the same manner as normalizing autocorrelations, we may use the normalization: $\mathbf{C}'_{xyz..}(k) = \mathbf{C}(k)_{xyz..}/\mathbf{C}(k)_{xxx..}$

[3]For example, correlation functions normalized to a maximum value of one are independent of the actual magnitude scaling of the input variables. Similarly, we would like to normalize the cumulants such that the cumulant slices will not change if the input variables are scaled in a similar manner.

Doing these normalizations will allow us to combine various cumulant estimators for the decision function.

D. The ICAIVS Algorithm

Based on the above results, it is now possible to derive an algorithm for selecting the desired inputs to a given model. Inputs can be selected based on the prespecified level of dependence allowed between input subsets and the desired output.

The algorithm can be described as follows.

1) For each cross-cumulant statistic, determine the average level of dependence implied by the magnitude of the statistic.
2) Compare each input in turn to this average.
3) Inputs which are significantly different from the average value are candidates for inputs to the model.

Hence, for ICA transformed inputs $\tilde{x}_{\mathbf{i}}\, i = 1,..,n$, considering the subsets of inputs, $\tilde{\mathbf{x}} = [x_1,\ldots,x_n, x_1x_2x_3,\ldots, x_1\cdots x_n]^{\mathrm{T}}$ where $\tilde{\mathbf{x}}_j$ is the jth element of $\tilde{\mathbf{x}}$,, hence we obtain the rule

$$\mathbf{S}_{xy}(i,k) = \begin{cases} 1 & \sum_i |\mathbf{C}_{xy}(i,k)| - E[\sum_i |\mathbf{C}(i,k)|] > K_{xy}(i,k) \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$\tilde{\mathbf{x}}_a(i,k) = \begin{cases} \tilde{\mathbf{x}}(i,k) & (\mathbf{S}_{xy}(k) \vee \mathbf{S}_{xxy}(k) \vee \mathbf{S}_{xxxy}(k)\ldots)_{i,k} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where

$\mathbf{C}_{xyj}(i,k)$	jth element of $\mathbf{C}_{xy}(i,k)$ at time $t-k$;
$\vee$	logical OR function;
$K_{xy}(i,k)$	threshold value chosen for each subset.

Selection of K_{xy} is aimed at choosing subsets of input variables that are indicated as statistically different from the other input variable subsets. Hence one method of doing this is to use the usual χ^2 test.

This rule means that if any of the cross cumulants for any given input subset are above a set threshold level, that particular input subset is deemed to be required for the model. The test is applied across all subsets of inputs, however in the equations above, only one input is actually shown.

E. Computational Complexity

For cross cumulants between p variables, we normally need to test for all cross-orders of cumulants up to the second power, which would be 2^p tests. For n input variables, this implies testing all possible subsets up to the set of n variables which leads to a total number of tests given by

$$N_{\mathrm{T}} = \sum_{i=1}^{n} \binom{n}{i} 2^i \tag{11}$$

$$= 3^n - 1 \tag{12}$$

While the test scales poorly, in practice, we may not need to test all possible combinations of inputs to establish whether a variable is required or not. Low cross-order terms can be tested initially and higher cross-orders can be restricted to variables found to be not required by the lower order tests. If the tests

of the lower orders suggest that p variables are not required we have only to test those cross cumulants in the next order in which these variables are involved. In other words, we can reduce all the tests which consists only of variables already known to be required. If we restrict the test to cumulants with second-order powers in any individual variable we have only to make

$$\left(\frac{n!}{r!n-r!}-\frac{n-p!}{r!n-p-r!}\right)2^{r+1} \tag{13}$$

tests in the order r of cross cumulants. The term 2^{r+1} is the number of cross cumulants to reach the second power.

Remarks:

1) The proposed algorithm relies on the existence of a statistically observable dependency between the true input variables and the output of the system.
2) One example of variables which may be observable through mixtures is noise. ICA has been shown to be in particular suitable for demixing noise from measurements. Our method is therefore very suitable for noisy data. We therefore expect this method to be useful in a wide domain of practical applications.
3) PCA is often used to select inputs, but this is not always useful, since the variance of a signal is not necessarily related to the importance of the variable. The features selected may have nothing to do with the problem.
4) In contrast to the use of PCA for input variable selection [15], [16], the variables we remove are those which are independent of the output, which is quite different from removing those with low variance.
5) Nonlinear functions may show zero second-order correlations and so it is necessary to use higher order statistics to estimate the dependence between variables. Hence, we assume non-Gaussian signals in order for the higher order cumulants to be nonzero.
6) The proposed algorithm is closed form and is guaranteed to give the dependence or otherwise of the input signals on the output variable.
7) Spurious correlations or dependencies may exist between unrelated variables and hence could lead to falsely included inputs, e.g., generated by coupled systems.
8) It is important to distinguish between a nonlinear mixing process and a nonlinear model. The method proposed here assumes linear mixing and hence it will not be strictly valid for the difficult case of nonlinear mixing. However, in many practical situations, linear mixing appears to be a reasonable approximation to what is observed. While the former presents some difficulties, the latter situation can be adequately dealt with by the model and is demonstrated through examples in Section III.

III. Simulation Examples

A. Example 1

In this example, we show the effect of using higher order cross cumulants as a means of detecting dependence among variables. Here 15 mutually independent binary independent, identically distributed (i.i.d.) signals were generated. Some of these inputs can be regarded as noise. Three signals, z_2, z_6 and z_9 were used as inputs to a system with output y. The nonlinear model is described by

$$y = z_2^3 + \cos(z_6) + 0.3\sin(z_9) \tag{14}$$

The input signals $\mathbf{z}$ are not observable directly, but passed through a random mixing matrix $\mathbf{A}$, such that we can observe $\mathbf{x} = \mathbf{Az}$. We apply the proposed algorithm for input variable selection to the observed variables $\mathbf{x}$ and to the signals recovered after application of ICA.

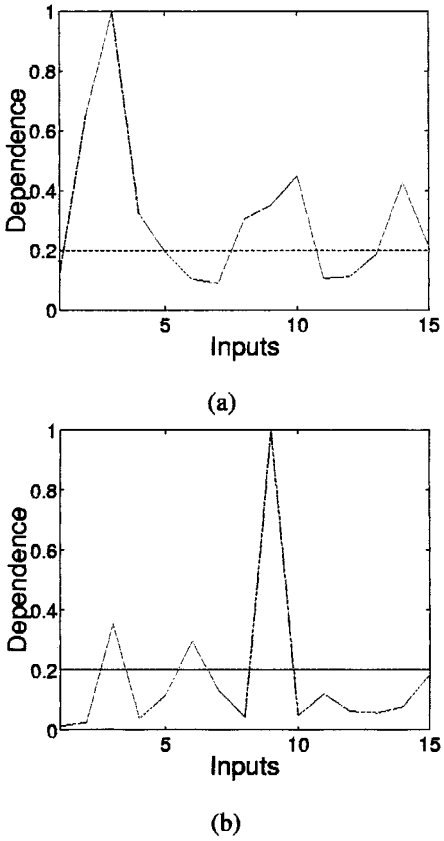

Fig. 1. Input variable selection Experiment 1 results: (a) without using ICA, seven inputs were selected, (b) using ICA the required inputs 2, 6, and 9 are correctly selected.

In Fig. 1(a), the results are shown for the case in which only the higher order statistics of cross-cumulants are used to determine the relevance of the inputs without first demixing the inputs. It is clear that when ICA is not used, the number of inputs are overestimated, viz., seven inputs were found to be relevant to the model.

In Fig. 1(b), the results are shown for the case of including the demixing before applying the statistical tests. Here, when ICA is used, the number and rank of inputs are estimated correctly. Examination of the normalized cross cumulants shows that the dependent inputs have markedly higher values than those which are independent of the output y.

These results are as expected. The first case has dependencies randomly spread across all observed inputs. Hence the number of inputs are overestimated. Note that the ranking can not be directly compared to the true rank which is obtained in the second case.

It is especially interesting to observe that the dependence is not always obvious with the second-order statistics, but the

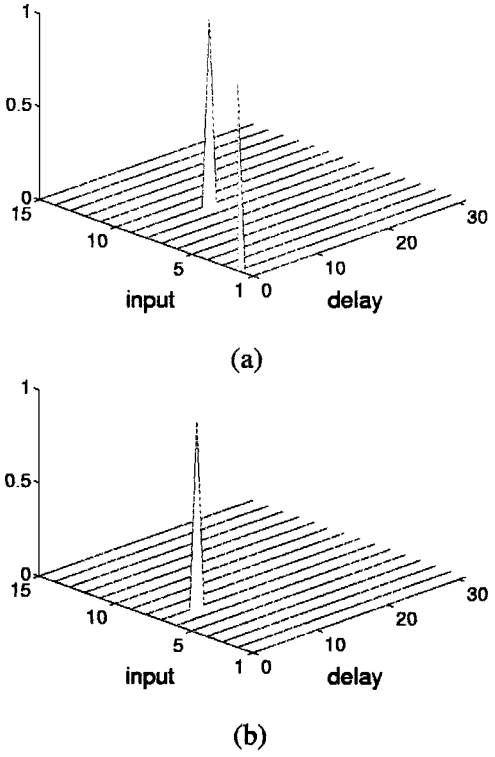

Fig. 2. Results for Experiment 2 on input variable selection. Here the cumulants are shown after thresholding according to the test described in the text. Shown are the thresholded cumulants (a) $c_{xy}(k)$ and (b) $c_{xxy}(0,k)$, $k = 1,\ldots,p$, respectively.

higher order cumulants serve a role in identifying all the required inputs.

B. Example 2

Suppose we observe a multivariate time series $\mathbf{x}(t) = [x_1(t)\ldots x_n(t)]$ consisting of dependent variables $\{x_i(t)\}$, and an output $y(t)$ from a system to be identified. For the purposes of this example, suppose $n = 15$ and the output is the result of a functional model given by

$$y(t) = F_o(z_2(t), z_7(t-5), z_9(t-11)). \tag{15}$$

where $F_o(a,b,c)$ is defined in this example, as

$$\begin{aligned} y =& (2a-1.6)^3 - 2a + (3b-3.5)^2 \\ & - 3b - (0.5c-0.8)^3 - 1.2c + 4. \end{aligned}$$

The input dependencies in $\mathbf{x}$ arise due to some true signals $\mathbf{z}(t) = [z_1(t)\ldots z_n(t)]$ becoming mixed, according to

$$\mathbf{x} = \mathbf{A}\mathbf{z}. \tag{16}$$

The results from this experiment are observed in Fig. 2 where the necessary inputs to the model are easily selected. Although not shown here, when ICA is not used, *all* inputs were identified in this case as being required. Thus, the algorithm successfully identified just the inputs required from the measured data. As indicated in Section II-E, when ICA is employed, it is often possible to reduce the number of tests required in terms of cumulant orders by first testing low cross-order terms initially.

IV. Conclusion

To effectively model and predict multivariate time series data it is important to use only inputs actually required and remove those inputs not required. If unrequired inputs are used significant problems can occur, especially in problems of high input dimensionality. Invariably it will be considerably more difficult to estimate a given model and the accuracy of the model will also suffer. Computational burden will also be increased dramatically due to the increased difficulty in learning. The problem of input variable selection normally assumes that it is possible to select the required optimal set of inputs directly from the set of measured data. However we have shown that this assumption is easily violated. In this case, overestimation of the number of inputs typically occurs [1].

In this paper, we proposed a new method for performing input variable selection which helps to solve the above problem. The method is based on the recently introduced method of independent component analysis. This approach permits a relatively straightforward statistical test to be derived for model free input variable selection. We applied the proposed algorithm to some examples which showed that it is capable of successfully isolating the inputs required for a model, even when the measured data itself is mixed and would normally lead to overestimating the number of inputs required. It is apparent that ICA provides a useful tool for accurately estimating the inputs required in building complex models, particularly with noisy input data as observed in many practical situations.

References

[1] B. V. Bonnlander and A. S. Weigend, "Selecting input variables using mutual information and nonparamteric density estimation," in *Proc. 1994 Int. Symp. Artificial Neural Networks (ISANN'94)*, Tainan, Taiwan, 1994, pp. 42–50.

[2] D. DeMers and G. Cottrell, "Nonlinear dimensionality reduction," in *Advances in Neural Information Processing Systems*. San Mateo, CA: Morgan Kaufmann, 1993, pp. 580–587.

[3] Y. Le Cun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, pp. 598–605.

[4] B. Hassibi and D. G. Stork, "Second-order derivatives for network pruning: Optimal brain surgeon," in *Advances in Neural Information Processing Systems 5*, S. J. Hanson, J. D. Cowan, and C. L. Giles, Eds. San Mateo, CA: Morgan Kaufmann, 1993, pp. 164–171.

[5] E. D. Karnin, "A simple procedure for pruning backpropagation trained neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 239–242, 1990.

[6] L. K. Hansen and C. E. Rasmussen, "Pruning from adaptive regularization," *Neural Comput.*, vol. 6, no. 6, pp. 1223–1232, 1994.

[7] C. Molina and M. Niranjan, "Pruning with replacement on limited resource allocating networks by f-projections," *Neural Comput.*, vol. 8, no. 4, pp. 855–868, 1996.

[8] C. L. Giles and C. W. Omlin, "Pruning recurrent neural networks for improved generalization performance," *IEEE Trans. Neural Networks*, vol. 5, pp. 848–851, Sept. 1994.

[9] G. Castellano, A. M. Fanelli, and M. Pelillo, "An iterative pruning algorithm for feedforward neural networks," *IEEE Trans. Neural Networks*, vol. 8, pp. 519–531, May 1997.

[10] M. Girolami, "A comparison of pruning techniques to enhance network generalization performance," in *Proc. 2nd NSYSN Workshop, Advanced Techniques, The Practical Applications of Neural Networks*, 1996, pp. 163–188.

[11] R. Reed, "Pruning algorithms — a survey," *IEEE Trans. Neural Networks*, vol. 4, pp. 740–747, Sept. 1993.

[12] C. Jutten and J. Herault, "Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, pp. 1–10, 1991.

[13] S. Amari and A. Cichocki, "Blind signal processing: Neural-network approaches," *Proc. IEEE* , 1998.
[14] A. Stuart and K. Ord, *Kendall's Advanced Theory of Statistics: Distribution Theory*, 6th ed. New York: Edward Arnold, 1994, vol. 1.
[15] A. U. Leen, T. K. Lenn, and J. E. Moody, "Fast pruning using principal components," in *Advances in Neural Information Processing Systems*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds. San Mateo, CA: Morgan Kaufmann, 1994, vol. 6, pp. 35–42.
[16] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," *Neural Comput.*, vol. 9, no. 7, pp. 1493–1516, 1997.