Portland State University

PDXScholar

7-2003

Using Reconstructability Analysis to Select Input Variables for Artificial Neural Networks

Stephen Shervais
Eastern Washington University

Martin Zwick
Portland State University, zwick@pdx.edu

Follow this and additional works at: https://pdxscholar.library.pdx.edu/sysc_fac

Part of the Logic and Foundations Commons, and the Systems Architecture Commons

Let us know how access to this document benefits you.

Using Reconstructability Analysis to Select Input Variables
for Artificial Neural Networks

Stephen Shervais
Eastern Washington University,
Cheney, WA 99004, USA
sshervais@ewu.edu

Martin Zwick
Portland State University,
Portland, OR 97207, USA
zwickm@pdx.edu

Abstract-- We demonstrate the use of Reconstructability Analysis to reduce the number of input variables for a neural network. Using the heart disease dataset we reduce the number of independent variables from 13 to two, with limited loss of accuracy compared with those of NNs using the full variable set. We also demonstrate that rule lookup tables obtained directly from the data for the RA models are almost as effective as NNs trained on model variables. This updated version corrects certain data errors in the original.

Index Terms— reconstructability analysis, artificial neural networks, information theory, OCCAM.

I. INTRODUCTION

An on-going problem when developing classifier systems is how to determine which features are worth paying attention to. The default approach, to include all variables and let the classifier sort them out, leads to computational intractability and to situations where related variables may end up fighting over what part of the variance each gets to explain. If the number of inputs can be reduced, either through domain knowledge or with the use of auxiliary tools, the likelihood of good performance is increased.

This paper uses a method called Reconstructability Analysis (RA) to reduce the number of variables used in an industry-standard classification problem. In this paper, RA is utilized to develop models which are simpler, i.e., have fewer variables, than the original problem, yet still capture most of the predictive information in the data. We then use these simpler models to analyze training and testing datasets for an artificial neural net, as well as to construct lookup tables specifying rules derived from the models. Related work on feature selection by RA methods has been reported by Lendaris, Shannon, and Zwick [11], Chambless and Scarborough [4], and Shannon and Zwick [12].

The rest of the paper is in five parts. In Section II, we provide a brief introduction to reconstructability analysis. In Section III, we describe the heart disease dataset. Section IV contains the procedures we used to build our training and testing datasets, while Section V presents our results for both the neural nets and the lookup tables. We finish with a discussion of the results in Section VI.

II. RECONSTRUCTABILITY ANALYSIS

Reconstructability analysis (RA) derives from Ashby [1], and was developed by Broekstra, Cavallo, Cellier, Conant, Jones, Klir, Krippendorff, and others; an extensive bibliography is available in [8], and a compact summary of RA may be found in [15][17]. RA resembles log-linear (LL) methods [2][9], used widely in the social sciences, and where RA and LL methodologies overlap they are equivalent [9][10]. In RA [7], a probability or frequency distribution or a set-theoretic relation is decomposed (compressed, simplified) into component distributions or relations. The most common application is the decomposition of frequency distributions, where RA does statistical analysis.

RA can model problems both where "independent variables" (inputs) and "dependent variables" (outputs) are distinguished (directed systems) and where this distinction is not made (neutral systems). In the present case, we have a directed system, with up to 13 independent variables A-M as inputs, and a single dependent variable, Z as the output. The goal, in our analysis, is to find some subset of the inputs that provides an acceptable level of prediction of the output. Since the information contained in a model is *not* the same as the classification rate, nor even a covariance measure, it is possible to obtain high classification rates with models that provide only limited information.

Consider a frequency distribution $f(A, B, C, Z)$ for a directed system, where A, B, and C are inputs and Z is an output. RA decomposes such distributions into models consisting of sets of projections, for example into $f_1(A,B,C)$, $f_2(A,B,Z)$ and $f_3(B,C,Z)$, written as the cyclic model ABC:ABZ:BCZ. Taken together, these three projections, two of which predict the output from the inputs, constitute a model of the data that is less complex (has fewer degrees of freedom) than the data. By maximum-entropy (uncertainty) composition of these projections, the model yields a *calculated* trivariate $ABCZ_{ABC:ABZ:BCZ}$ distribution (the subscripts show the model used), which may differ from the *observed* ABCZ data. Such a model may be used for prediction, and may be assessed by its %Uncertainty Reduction, 100. $[H(Z) - H_m(Z|ABC)]$ / $H(Z)$, where H is Shannon entropy, and

$H_m(Z|ABC)$ is the conditional entropy of the output, knowing the inputs, for model m.

A simpler class of RA models involves only a single "predicting component" (a component including the output), and these models have no loops. For example, ABC:ABZ says that Z is predicted by A and B. Models of this sort select a *subset* of inputs as predictors from the full set of inputs specified by the first component. It is such models that are used in this paper for variable reduction (feature selection). The more complex, multiple-predicting-component models can also be used to predict the output, as discussed briefly at the end of Section V, or to prestructure a neural net with less than full connectivity ([5] and papers cited therein).

Calculations for this paper were made using the RA software programs developed at Portland State University, now integrated into the package OCCAM (for the principle of parsimony and as an acronym for "Organizational Complexity Computation And Modeling"). The earliest of these programs was developed by Zwick and Hosseini [6]; a list of recent RA papers of the PSU group is given in [14][16].

III.   HEART DISEASE DATASET

The University of California at Irvine maintains a repository of machine learning databases, including a collection of data used for predicting the presence or absence of heart disease. The dataset we used is a cleaned version of the UCI Cleveland heart disease dataset, obtained from the University of Porto, in Portugal.

A.   Description

The dataset has 270 records, with 13 independent variables (a subset of the original 75 variables) and one dependent variable. The 13 independent variables include 5 continuous variables (A,D,E,H,J), one ordered variable (K), one integer (L), three binaries (B,F,I), and three multi-value nominal (E,C,M). The dependent variable originally coded for five levels of disease, including no disease. In keeping with standard practice, the processed dataset we used simply reports the presence or absence of heart disease.

Looking ahead to the key variables found using RA, variable C represents four levels of chest pain, variable L represents the number of major blood vessels colored by fluoroscopy (up to three), and variable M represents three classes of heart defects detected in a thallium imaging test.

B.   Data Extraction

In preliminary work with and research on the dataset, we noticed there was a wide range of success in the application of different tools [3][13]. *A priori* we attributed that to the use of different partitionings of the dataset, as well as to variations in the quality of the tools. To control for this, we partitioned the 270-record dataset into five different training/testing sets on an 80/20 basis, with 216 records in the first, and 54 in the second. We did this by assigning each record randomly to either the training or testing sets, with probabilities 0.8/0.2, dropping any dataset that was not partitioned 216/54, and repeating the process until we had five datasets that matched our requirements. The majority class of the training sets was *disease absent*, and a naïve predictor using this would have a 50.6% success rate in classifying the test data.

IV.   PROCEDURE

The first step in reconstructability analysis is to bin any continuous variables in the dataset. There are five such variables, and they were each binned into four bins of approximately equal frequency. Next, each of the five dataset extracts, with binned variables, was processed by the OCCAM software. Table I shows the best three of the two-input models for each experimental dataset. The best model for the *whole* 270-record dataset is CMZ; however two other models, LMZ and CLZ score consistently better in the 216-record training set extracts. The second column under each experiment shows the uncertainty reduction of the associated model. The cardinalities of the models we will thus consider, namely CMZ, LMZ, and CLZ, are 24, 24, and 32, respectively.

TABLE I.
THE THREE HIGHEST-SCORING MODELS FROM EACH OF
FIVE PARTITIONINGS OF THE DATASET.

Experiment	Model	Uncertainty Reduction
	LMZ	39.1%
1	**CMZ**	35.2%
	JMZ	34.4%
	LMZ	34.8%
2	KLZ	33.8%
	CKZ	32.4%
	LMZ	41.8%
3	**CMZ**	41.3%
	CLZ	38.1%
	CLZ	40.7%
4	**CMZ**	38.0%
	CJZ	37.7%
	CLZ	37.6%
5	**CMZ**	37.5%
	LMZ	37.4%

Note that in experiment 2, the CMZ model did not make the top three.

The five datasets were reduced to just the variables in the high scoring models. Generically, we shall refer to these as the *primary* models, to differentiate them from the *CMZ* model, known to be best on the full dataset. The variables associated with the primary models were used (a) to create rule sets for lookup tables and (b) for training and testing datasets for the neural networks. The same 54-record testing sets were used to test the classification abilities of each of the approaches.

The rule sets were constructed by counting the instances of each outcome (1 or 0) in the output variable for a given set of values in the input variables and assigning a rule based on the majority of the outcomes. Table II shows the process and resulting rule set for the CMZ model, using Experiment 1 data. In the training dataset for CMZ there were nine instances where $C = 1$ and $M = 0$. In seven of those instances, the value of Z was 0 (no disease) and in two, the value of Z was 1 (disease present). The rule therefore assigns all future (testing) instances of $C = 1$, $M = 0$ to the *no disease* category. Since the majority of the training set showed $Z = 0$, any instance of a tie ($C = 3$, $M = 2$, for example) was assigned to the *no disease* class, as was any input variable combination that was empty in the training set. (We did not attempt to use proximity to other input states to break ties or resolve sampling zeroes in the training set.)

TABLE II.
RULE SET FOR MODEL CMZ,
BASED ON EXPERIMENT 1 DATA.

C	M	Rule	Z_0	Z_1	Score
1	0	0	7	2	7
1	1	0	2	0	2
1	2	0	1	1	1
2	0	0	25	1	25
2	1	0	1	1	1
2	2	0	3	3	3
3	0	0	41	5	41
3	1	1	0	1	1
3	2	0	9	9	9
4	0	0	24	17	24
4	1	1	3	4	4
4	2	1	6	50	50

Columns C and M show the different values possible for those variables. Columns Z0 and Z1 count the number of outcomes for that CM combination. If Z0 is higher, then the Rule for that combination of input values is set to zero. If there is a tie, then the Rule defaults to zero. The Score column counts how many of the Z-values each particular rule correctly captures. The total score for CMZ was 168 and since the sample size is 216, the % correct of this rule set is 77.8%.

For the NN version of the dataset, the original (unbinned) variables were normalized so their values all lay between one and zero. The NN used (Figure 1.) had two input nodes, three hidden nodes, one bias node, and one output node. The hidden and output nodes used a log-sigmoid transfer function with continuous outputs that range from 0 to 1. The input nodes connected only to the hidden layer.
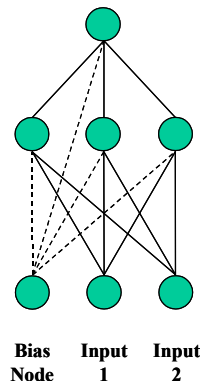

FIGURE 1. NEURAL NET STRUCTURE.
The inputs are either chest pain and thallium imaging results (CM), chest pain and fluoroscopy results (CL), or fluoroscopy results and thallium results (LM).

During training, the errors were computed based on the continuous outputs. For testing purposes, since the object was classification, the output was forced to 1 (if $>= 0.5$) or to 0 (if < 0.5). The NNs were trained for ten complete 216-record training events broken into 16 record epochs. Learning rate was 0.8 and momentum was 0.6 throughout the process. For each training set, the NN was initialized 100 times and the results saved and averaged.

One of the purposes of reducing the number of variables is to be able to lessen the computational load on the NN. Dropping from thirteen inputs (and 225 weights) to two inputs (and 13 weights) provided a roughly six fold improvement in training time, from three minutes to thirty seconds on an 800MHz Pentium.

V.   RESULTS

The results are shown in Table III and IV. Classification performance of a set of model-based NNs is shown in Table III, in the sense of percentage of test records correctly classified. Three NNs were used for each experimental dataset: one based on the two input variables (CM) identified in the known-best model, one based on the two input variables (LM or CL) found in the best-available (primary) model, and one based on all 13 input variables (full). The results shown in Table III show that for a given NN architecture and training regime the full 13-input NN provides the best classification accuracy, 81.6%, an increase of 31% from the naïve predictor mentioned at the end of Section III. Reducing the number of inputs from 13 to two diminishes predictive accuracy by 6.4% to 75.2%, an increase of 24.6% over naïve prediction. This means

that 79% of the information available to the NN is available in the two best inputs.

In paired t-Tests, displayed in Table V, ahead, the improved results for the full NN model were statistically significant at a 0.01 level. The performance of the primary model NNs were not significantly different (at the 0.10 level) from NNs based on the known best model.

TABLE III.
NN-BASED CLASSIFICATION PERFORMANCE OF OCCAM-DERIVED MODELS.

EXPERIMENT	PRIMARY MODEL	CMZ NN	PRIMARY NN	FULL NN
1	LMZ	75.5%	74.7%	83.0%
2	LMZ	82.3%	82.5%	85.1%
3	LMZ	69.1%	71.5%	76.5%
4	CLZ	75.2%	72.1%	81.5%
5	CLZ	77.5%	78.2%	81.7%
Mean		75.9%	75.2%	81.6%
Std Dev		4.75%	4.6%	3.2%

Performance on the 54-record testing sets by the neural nets. Primary models were the ones with the highest RA information score on the five training extracts of the data; CMZ was the best model on the full data. The Full NN used all 13 variables as inputs.

Looking now at the performance of the rulesets, obtained directly from the data for the variables selected by RA (Table IV) the best-model ruleset did only slightly better than the primary model rulesets; this difference was significant at a 0.10 level (Table V). For the primary models, the difference between the NN results, 75.2 ± 4.6%, and the ruleset results, 72.2 ± 5.4%, was also small

TABLE IV.
RULE-BASED CLASSIFICATION PERFORMANCE OF OCCAM-DERIVED MODELS.

EXPERIMENT	PRIMARY MODEL	CMZ RULES	PRIMARY RULES
1	LMZ	77.8%	74.1%
2	LMZ	81.5%	79.6%
3	LMZ	70.4%	70.4%
4	CLZ	68.5%	64.8%
5	CLZ	72.2%	72.2%
Mean		74.1%	72.2%
Std Dev		5.1%	5.4%

Performance of the rule-lookup tables of the primary models and of the known-best CMZ model on the 54-record testing sets.

but significant at a 0.005 level. The differences between the two-input NN results and those of the best-model (CMZ) ruleset were not statistically significant.

The performance of all the tools varied considerably across the different experimental datasets, with standard deviations ranging from 3.2 to 5.4 percentage points. This supports our contention that experiments with a single partitioning of a small dataset such as the heart disease data cannot be trusted to give an accurate portrayal of a classification tool's effectiveness.

TABLE V.
RESULTS OF T-TEST ON THE DIFFERENT CLASSIFICATION TOOLS.

	CMZ MODEL RULES	PRIMARY MODEL RULES	CMZ MODEL NN	PRIMARY MODEL NN
PRIMARY MODEL RULES	0.09			
CMZ MODEL NN	0.36	0.14		
PRIMARY MODEL NN	0.32	0.05	0.90	
Full NN	0.01	0.01	0.004	0.012

Comparisons of NN and rulesets are discussed in the text. Table entries are t-Test p-values and represent probability there is no significant performance difference between the classifiers represented on the two axes.

Lastly, it should be noted that because RA is used in this paper to select subsets of IV predictors for NNs, only the simplest RA models are used, namely models without loops. However, past experience with RA models strongly suggests that rulesets derived from models with loops can have greater predictive power. We have examined this possibility by considering rulesets from the model CZ:JZ:LZ:MZ, which in terms of degrees of freedom is simpler than CLZ and as simple as LMZ and CMZ. This model achieves a 73.0 ± 7.7 %correct on the five experiments, which is not statistically distinguishable from the 75.2 ± 4.6 %correct score of the primary NNs. The examination of such loop-containing RA models is still underway, and will be more fully reported in a later communication. In this paper we are more concerned with using RA to simplify NNs rather than finding maximally predictive RA models.

VI.   DISCUSSION

We have shown that applying the simplest form of reconstructability analysis (using loopless single-predicting-component models) will allow us to reduce the number of variables in a standard problem to a small subset of the original, and that this reduction allows the creation of simple NN architectures that have most of the predictive power of maximally complex NNs. Since a

simpler NN that can learn the training set is, in theory, more likely to generalize well compared to a more complex NN of equal performance, it is to be preferred. Moreover, we also find that predicting the output with a simple and completely transparent look-up table obtained directly from the data performs almost as well as NNs trained on the same data subsets. Why the primary NNs did better than the primary rules may involve the small sample sizes of the training sets and/or the NN use of metric information. For example variable L is a quantitative variable, C is an ordinal variable, and M could perhaps be ordinal if two of its values were swapped. The NN may make use of some of this information, not exploitable by a rule table. These possibilities are under investigation. Whether the NN advantage would hold as well for larger datasets also remains to be studied.

REFERENCES

[1] Ashby, W. R. 1964. "Constraint Analysis of Many-Dimensional Relations." *General Systems Yearbook*, 9, pp. 99-105.

[2] Bishop, Y., S. Feinberg, and P. Holland, 1978. *Discrete Multivariate Analysis*. MIT Press, Cambridge.

[3] Chakraborty, S., Pal, K., and Pal, N. (2002) "A neuro-fuzzy framework for inferencing", *Neural Networks*, 15, pp. 247-261.

[4] Chambless, B., and Scarborough, D. (2001). "Information-Theoretic Feature Selection for a Neural Behavioral Model", International Joint Conference on Neural Networks (IJCNN), Washington D.C. http://www.sysc.pdx.edu/download/papers/feature.pdf

[5] Chambless, B., Lendaris, G., and Zwick, M. (2001). "An Information Theoretic Methodology for Prestructuring Neural Networks", International Joint Conference on Neural Networks (IJCNN), Washington D.C. http://www.sysc.pdx.edu/download/papers/nnra.pdf

[6] Hosseini, J.C., R. R. Harmon, and M. Zwick, "Segment Congruence Analysis Via Information Theory." *Proceedings, International Society for General Systems Research*, Philadelphia, PA, May 1986, pp. G62 - G77. http://www.sysc.pdx.edu/download/papers/inftheoretic.pdf

[7] Klir, G. 1985. *The Architecture of Systems Problem Solving*. Plenum Press, New York.

[8] Klir, G. 1986. "Reconstructability Analysis: An Offspring of Ashby's Constraint Theory." *Systems Research*, 3 (4), pp. 267-271.

[9] Knoke, D. and P.J. Burke1980. *Log-Linear Models*. (Quantitative Applications in the Social Sciences Monograph # 20). Sage, Beverly Hills.

[10] Krippendorff, K. 1986. "Information Theory: Structural Models for Qualitative Data." (Quantitative Applications in the Social Sciences #62). Sage, Beverly Hills.

[11] Lendaris, G., Shannon, M., and Zwick, M. (1999). "Prestructuring Neural Networks for Pattern Recognition Using Extended Dependency Analysis", invited paper, Applications and Science of Computational Intelligence II AeroSense'99, Orlando, FL, SPIE. http://www.sysc.pdx.edu/download/papers/99spie.pdf

[12] Shannon T., and Zwick M. (2002) "Directed Dependency Analysis for Data Mining". 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop, Pittsburgh. http://www.sysc.pdx.edu/download/papers/99spie.pdf

[13] Skalak, D. (1995) *Prototype Selection for Composite Nearest Neighbor Classifiers*, CMPSCI Technical Report 95-74, University of Massachusetts, Amherst.

[14] Willett,K., and Zwick, M. (2002). "A Software Architecture for Reconstructability Analysis." Proceedings of 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop, Pittsburgh, March 24-26.

[15] Zwick, M. (2001a). "Wholes and Parts in General Systems Methodology." In: *The Character Concept in Evolutionary Biology*, edited by Gunter Wagner. Academic Press, New York, 2001a, pp. 237-256. http://www.sysc.pdx.edu/download/papers/wholesg.pdf

[16] Zwick, M. (2001b). "Discrete Multivariate Modeling": http://www.sysc.pdx.edu/res_struct.html

[17] Zwick, M. (2002). "An Overview of Reconstructability Analysis." 12th International World Organization of Systems and Cybernetics and 4th International Institute for General Systems Studies Workshop, Pittsburgh. http://www.sysc.pdx.edu/download/papers/ldlpitf.pdf