Systems Analysis Modelling Simulation
Vol. 43, No. 10, October 2003, pp. 1311-1319

Taylor & Francis
Taylor & Francis Group

APPLYING GMDH ALGORITHM TO EXTRACT RULES FROM EXAMPLES

KOJI FUJIMOTO* and SAMPEI NAKABAYASHI

Financial Engineering Group, Inc., Akasaka Yamada Bldg. 2-21-8, Akasaka, Minato-ku, Tokyo, 107-0052, Japan

(Received 13 August 2001)

This article reports a new approach to rule extraction method by using Group Method of Data Handling (GMDH) Algorithm in Data Mining area. The advantages of this method are (1) it accepts both categorical and continuous data at the same time, and (2) rules can be extracted easily from the generated model.

We applied GMDH Algorithm to categorical data set of US congress voting records to extract rules. The correction rate of GMDH rules was 97.3% - higher than Tsukimoto's method of rule extraction from Back-propagation neural network (81.0%). It was also higher than 97.0% of C4.5.

Keywords: GMDH; Rule extraction; Data mining; Neural network; Machine learning; C4.5

RELATED WORKS OF RULE EXTRACTION FROM ARTIFICIAL NEURAL NETWORKS

In this several years, many approaches of rule extraction from artificial neural networks were reported. Tsukimoto developed a structural analysis of neural networks using multi-linear functions and showed his new method of IF-THEN rule extraction from back propagation neural networks [1]. Also Gallant's method [2], SUBSET [3] and KT Algorithm [4] were reported as rule generation method from learning neural networks. The main ideas were: (1) using back propagation neural networks; and (2) convert one neuron unit into one IF-THEN rule.

Tsukimoto's Rule Extraction Method

In this section, we briefly discuss Tsukimoto's rule extraction method that is typical of rule extraction from neural network as the related work of our research.

In Fig. 1, the unit has five inputs as x_1 to,x_5 and one output y.

Each input variables take value of zero or one and the weights are 0.6, 0.2, 0.2, 0.2 and 0.05. The threshold 0.5 and transformation function calculate real number between 0 and 1 and set it to output variable y.

*Corresponding author. *E-mail: kfujim@feg.co.jp

ISSN 0232-9298 print: ISSN 1029-4902 online © 2003 Taylor & Francis Ltd
DOI: 10.1080/0232929032000115047

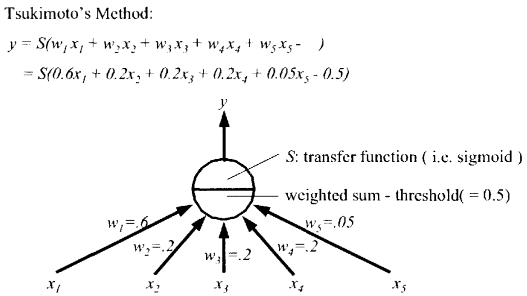

FIGURE 1 A back propagation neural network unit.

The first step is the single variable searching phase for IF condition. For a start, we make the single variable term list like x_1, $x_2, \ldots,$ and x_5. And calculate output value where each term is true. If the value is greater than 0.5, the test is accepted and we know the term is included in IF condition. For x_1, we calculate the output value where x_1 equal to 1 and x_2, $x_3, \ldots,$ and x_5 equal to zero. The weighted sum is 0.6 and the threshold is 0.5, so the output value becomes greater than 0.5 and the test is accepted. For x_2, the output value is less than 0.5 so, the test is declined. Also for x_3, x_4 and x_5, each test is declined.

Therefore only the test for x_1 were accepted and we found term x_1 includes the IF condition among the single variable terms.

The next step is the two-variables term searching phase. This time, we have to make a two-variable pair list among input variables of x_2, $x_3, \ldots,$ and x_5 except x_1. Because x_1 was already selected in the previous process. We do not have to check the term including variable x\.

For test of x_2x_3 variable pair, the weighted sum is 0.2 + 0.2 when x_2 and x_3 are true. However the output is less than 0.5 and the test is declined. Also all the other pairs failed in the tests and in this case, no two-variable terms are selected.

In three-variables term searching phase, we prepare three-variable tuple list among input variables of x_2, $x_3, \ldots,$ and x_5 except x_1 again.

For test of $x_2x_3x_4$ variable tuple, the weighted sum is 0.2 + 0.2 + 0.2 when x_2, x_3 and x_4 are true. The output is greater than 0.5, so the test is accepted. The other tuples failed in tests and we found terms x_2, x_3, x_4 conjunction term included in IF condition.

Tsukimoto's multi-variable term completed searching in three or several variable tuple levels. The reason why, he said that the high order conjunction term is too difficult for humans to understand.

Now we know single term x_1 and conjunction term $x_2x_3x_4$ includes the IF condition.

Therefore we get the rule of IF x_1 or x_2 and x_3 and x_4 the y equal to 1. And else y equal to zero.

This was an example of Tsukimoto's rule extraction from Back-propagation neural networks.

GROUP METHOD OF DATA HANDLING ALGORITHM

The group method of data handling (GMDH) was introduced by Ivakhnenko in 1966 [5] as an inductive learning algorithm for complex systems modeling (i.e. pattern recognition, prediction modeling for ecological and economical fields, etc.).

GMDH Neural Network Structure

The GMDH model has a forward multi-layer neural network structure. Each layer consists of one or more units, two input one output arcs.

Each unit corresponds to Ivakhnenko polynomial form [6]

$$z = a + bx + cy + dx^2 + exy + fy^2 \tag{1a}$$

or

$$z = a + bx + cy + dxy \tag{1b}$$

with two input variables x and y, an output variable z, and parameters $a, b, \ldots, f$. Figure 2 illustrates a typical multi-layer network structure.

Inductive Learning Algorithm

The basic technique of GMDH learning algorithm is a self-organization method. It fundamentally consists of the following steps:

(1) Given a learning data sample including a dependent variable y and independent variables $x_1, x_2, \ldots, x_m$; split the sample into a training set and a checking set.
(2) Feed the input data of m input variables and generate combination $(m,2)$ units from every two variable pairs at the first layer.
(3) Estimate the weights of all units (a to f in formula (la) or (1b)) using training set. In this report, stepwise regression method with formula (lb) was employed.
(4) Compute mean square error between y and prediction of each unit using checking data.
(5) Sort out the unit by mean square error and eliminate bad units.
(6) Set the prediction of units in the first layer to new input variables for the next layer, and build up a multi-layer structure by applying Steps (2) (5).
(7) When the mean square error become larger than that of the previous layer, stop adding layers and choose the minimum mean square error unit in the highest layer as the final model output.

Steps (4) and (5) describe an important and basic techniques of GMDH algorithm. It is called regularity criteria and leads to achieving the minimum at Step (7) [7].

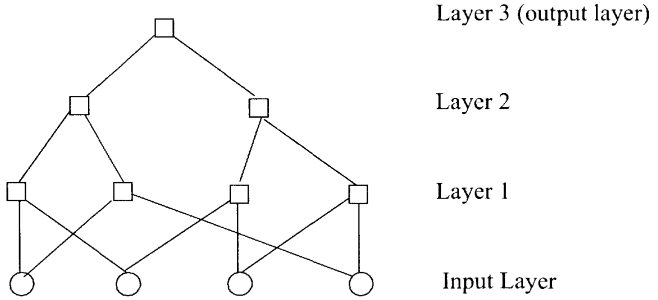

FIGURE 2 GMDH neural network structure with 4 inputs (○) and 7 units (□) in three layers.

RULE EXTRACTION METHOD FROM GMDH NEURAL NETWORK

GMDH Network Separation into Sub-networks

Like as shown in Fig. 3, some GMDH networks can be large. When it has around 7 or more input variables, we have to separate it into several sub-networks with 6 or less input variables. The rule extraction process becomes too complex with many input variables.

From here, the author will focus on the rule extraction method for a small GMDH network or separated sub-network.

Rule Extraction from a GMDH Sub-network

Using GMDH network structure developed from the data set of nominal dependent variable y and independent variables x_i, the following model is obtained:

Output Layer:

$$\hat{y} = a^{(G)} + b^{(G)}x_1^{(G-1)} + c^{(G)}x_2^{(G-1)} + d^{(G)}x_1^{(G-1)}x_2^{(G-1)} \tag{2a}$$

Hidden Layers:

$$x_i^{(k)} = a_i^{(k)} + b_i^{(k)}x_m^{(k-1)} + c_i^{(k)}x_n^{(k-1)} + d_i^{(k)}x_m^{(k-1)}x_n^{(k-1)} \quad (k = 1, \ldots, G-1) \tag{2b}$$

Input Layer:

$$x_i^{(1)} = a_i^{(1)} + b_i^{(1)}x_m^{(0)} + c_i^{(1)}x_n^{(0)} + d_i^{(1)}x_m^{(0)}x_n^{(0)} \tag{2c}$$

where G is the maximum layer, $x_i^{(k)}$ and $a_i^{(k)}, \ldots, d_i^{(k)}$ are the output and parameters of unit i in layer k $(k— 1, \ldots, G—1)$, and $x_n^{(0)}$ is input variable n.

Let us generate vectors $X_i = (x_1, x_2, \ldots, x_M)$, $i = 1, \ldots, 2^M$, $x_j = 0$ or 1 where M is the number of selected input variables in formula (2c). For each X_i, corresponding estimate y of $\hat{y}_i$ is computed using formulas ((2a), (2b) and (2c)). Now we get the following gain chart sorted and re-indexed by $\hat{y}_i$ (see Table I).

Accumulated summation S_i is defined as:

$$S_i = \sum_{k=1}^{i} n_{0,k} + \sum_{k=i+1}^{2^M} n_{1,k}. \tag{3}$$

We can get index number i that satisfies $S_i = \max\{S_k | k = 1, \ldots, 2^M\}$, and rule set as the followings:

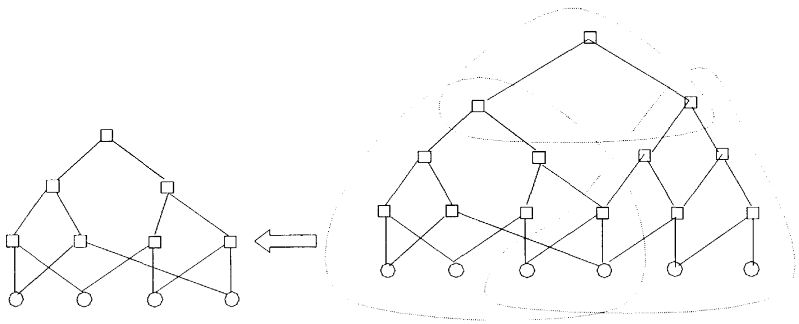

FIGURE 3 GMDH Network separation into sub-networks.

TABLE 1 Gain chart

X_1	$\hat{y}_1$	$n_{0,1}$	$n_{1,1}$	S_1
X_2	$\hat{y}_2$	$n_{0,2}$	$n_{1,2}$	S_2
:	:	:	:	:
X_i	$\hat{y}_i$	$n_{0,i}$	$n_{1,i}$	S_i
:	:	:	:	:
x_{2^M}	$\hat{y}_{2^M}$	$n_{0,2^M}$	$n_{1,2^M}$	S_{2^M}

$n_{0,i}$ is the number of observations satisfying the condition $y=0$ for X_i and $n_{1,i}$ is defined in the same manner for $y=1$.

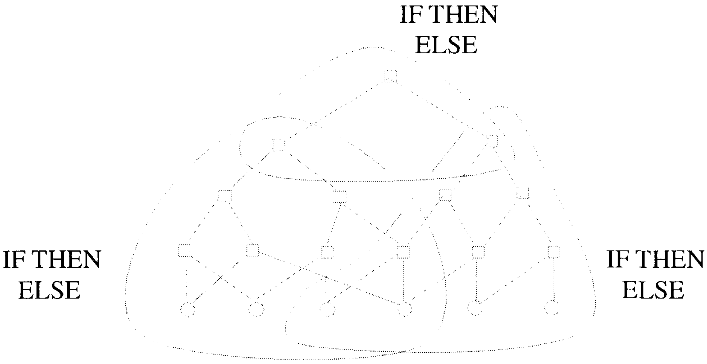

FIGURE 4 Integration of sub-rules generated from **sub-GMDH** networks.

$$\text{IF } X_i \vee X_2 \vee \ldots \vee X_{i-1} \quad \text{THEN} \quad y = 0.$$
$$\text{ELSE } y = 1.$$

Integration of Sub-rules

After rule extraction for each sub-network of GMDH, several sub-rules are obtained. The action part of the sub-rule for the lower sub-network is corresponds to the condition part of the sub-rule for the upper sub-network. Then, by integrating all of the sub-rules, we can obtain a rule set for the whole GMDH network (See Fig. 4).

CASE STUDY OF US CONGRESSIONAL VOTING RECORDS

We use the data on US Congressional voting records obtained from the University of California at Irvine Machine Learning Database [8] for testing our approach. After eliminate records containing missing fields from this data, we obtain 232 observations with one nominal dependent variable and 16 nominal independent variables. We applied the GMDH rule extraction method for this data and compared the result was compared with Tsukimoto's method (rule extraction from Back-propagation neural network [1] and C4.5 [9].

Data Specification

This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes. The data specification is shown below, Data Set Name: 1984 United States Congressional Voting Records

Data Size: 232 observations. Original Data Set has 435 Congressmen's records. 203 records in 435 those have missing values were eliminated from the data.

Nominal dependent variable y: (y) Party Name; classes are $1 =$ democrat and $0 =$ republican.

Nominal independent variables of 16 votes $x_1, x_2, \ldots, x_{16}$ are: (x_1) handicapped-infants; (x_2) water-project-cost-sharing; (x_3) adoption-of-the-budget-resolution; (x_4) physician-fce-freeze; (x_5) el-salvador-aid; (x_6)religious-groups-in-schools; (x_7)anti-satellite-test-ban; (x_8) aid-to-nicaraguan-contras; (x_9) mx-missile; (x_{10}) Immigration; (x_{11}) synfuels-corporation-cutback; (x_{12}) education-speding; (x_{13}) superfund-right-to-sue; (x_{14}) crime; (x_{15})duty-free-exports; and (x_{16})export-administration-act-south-africa. Values are $1 =$ Agree and $0 =$ Disagree for each votes.

Prediction Model of GMDH

Tn order to predict party class from the 16 voting values, we set them into output and input variables of GMDH input layer, and developed the prediction model.

Group method of data handling added layers twice and the learning Algorithm stopped. Figure 5 shows GMDH neural network structure of the voting data set.

Unit $x_i^{(2)}$ is selected to estimator of y because it has minimum mean square error among 11 units of output layer. The created prediction model of 2-layer with 3 units is as follows:

$$\hat{y} = x_1^{(2)} = 0.00320 + 0.83260 x_1^{(1)} + 0.19798 x_1^{(1)} x_{10}^{(1)}. \tag{4a}$$

$$x_1^{(1)} = 0.01667 + 0.98333 x_4 - 0.20000 x_4 x_{11}. \tag{4b}$$

$$x_{10}^{(1)} = 0.07407 + 0.86470 x_5 - 0.55416 x_3 x_5. \tag{4c}$$

where x_3, x_4, x_5 and x_{11} are input variables of attribute number $3, 4, 5$ and 11. The model has three unit outputs as $x_1^{(1)}$, $x_{10}^{(1)}$ in layer 1 and $\hat{y} = x_1^{(2)}$ layer 2. Thus gain chart shown in Table II was computed using the model formulas.

The column $\hat{y}_i$ is estimated value of y calculated by using the above equations ((4a), (4b) and (4c)) for ith low of $X_i = (x_3, x_4, x_5, x_{11})_i$ and the table are sorted by $\hat{y}_i$. The

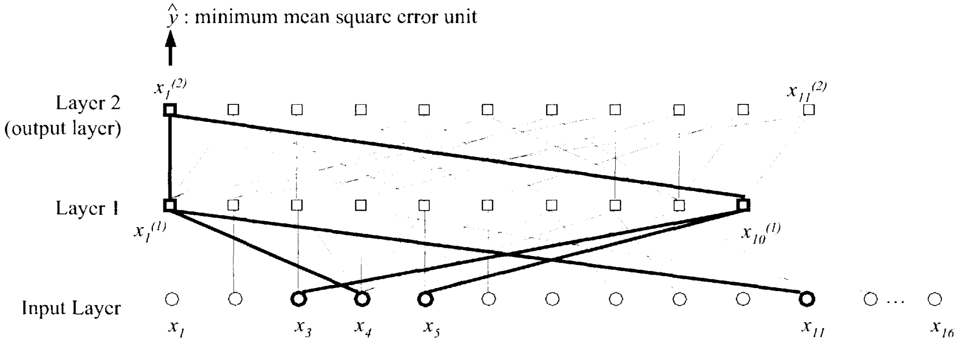

FIGURE 5 **GMDH** neural network for the voting records.

TABLE II Gain chart for Voting records sorted by $\hat{}yi$

I	x_3	x_4	x_5	x_5	$\hat{y}_i$	*Diff.*	*Frequency*		*# of Correct*			*Correction Rate* (%)
							$n_{o,i}$	$n_{1,i}$	$\sum n_{0,k}$	$\sum n_{1,k}$	S_i	
1	1	0	0	0	0.017	−52	52	0	52	108	160 (S_1)	69.0
2	1	0	0	1	0.017	−38	38	0	90	108	198 (S_2)	85.3
3	0	0	0	1	0.017	−5	5	0	95	108	203	87.5
4	0	0	0	0	0.017	−2	3	1	98	107	205	88.4
5	1	0	1	1	0.019	−9	9	0	107	107	214	92.2
6	1	0	1	0	0.019	−4	4	0	111	107	218	94.0
7	0	0	1	1	0.020	−5	5	0	116	107	223	96.1
8	0	0	1	0	0.020	−2	2	0	118	107	225	97.0
9	0	1	0	1	0.680	−1	1	0	119	107	226 (S_9)	97.4
10	1	1	0	1	0.681	1	0	1	119	106	225	97.0
11	1	1	1	1	0.730	−1	3	2	122	104	226 (S_{11})	97.4
12	0	1	1	1	0.818	12	2	14	124	90	214	92.2
13	0	1	0	0	0.850	1	0	1	124	89	213	91.8
14	1	1	0	0	0.850	2	0	2	124	87	211	90.9
15	1	1	1	0	0.911	12	0	12	124	75	199	85.8
$2^4 = 16$	0	1	1	0	1.021	75	0	75	124	0	124 (S_{16})	53.4

column $n_{0,i}$ is frequency of $y = 0$ for X_i and $n_{1,i}$ is frequency of $y = 1$ for X_i. Diff. is deference between $n_{1,i}$ and $n_{0,i}$. $\sum n_{0,k}$ is summation of $n_{0,1}$ to $n_{0,i}$ and $\sum n_{1,k}$ is summation of $n_{1,i+1}$ to $n_{1,16}$. S_i is sum of $\sum n_{0,k}$ and $\sum n_{1,k}$ defined as (3).

The chart shows the accumulation S_9 (or S_{11}) is maximum and marks the highest correction rate (97.4%). Then combinations $X_1 = (x_3, x_4, x_5, x_{11}) = (1,0,0,0)$, $X_2 = (1,0,0,1), \ldots, X_9 = (0,1,0,1)$ are employed to "Republican" rules. Therefore we get the following rules for the prediction.

IF $\sim x_4 \quad \vee (\sim x_3 x_4 \sim x_5 x_{11})$ THEN $y = 0$: Republican.
ELSE $y = 1$: Democrat.
Correction rate 97.4%

Comparison with C4.5 and Back Propagation Neutral Network

Here the authors present a comparison of our method with C4.5 [9] and Tsukimoto's new method of rule extraction from Back Propagation Neural networks using multilinear functions [1].

Method: Tsukimoto's rule extraction from NN.
Rule for republican $\sim x_4((x_3 \sim x_{11})\ v(x_3 x_{10})) \vee (\sim x_{10} \sim x_{11}))$
Correction rate: 81.0%

Method: C4.5
Rule for republican: $\sim x_4 \vee (x_3\ x_{11})$.
Correction rate: 97.0%

Method: Rule extraction from GMDH
Rule for republican: $\sim x_4 \vee (\sim x_3 x_4 \sim x_5 x_{11})$.
Correction rate: 97.4%

where $\sim$: NOT operator.

In experience with voting data, our method of rule extraction from GMDH is the highest correction rate among the three methods.

CONCLUSIONS

We introduced a method of rule extraction by applying GMDH Algorithm. The above result shows its effectiveness in knowledge discovery and data mining in large scale databases. However follow up research and expansion of the rule extraction method for a mixture of nominal, ordinal and continuous data will be required.

Ackno wledgments

The authors especially thank Dr. Alexy G. Ivakhnenko who advised the GMDH Algorithm. They would also like to thank Hiroshi Tsukimoto who provided detail information about his study, UCI Machine Learning Database project team for providing the test data and Dr. Bipin Indurkhya for reviewing the report.

References

[1] H. Tsukimoto, N. Shimogori and B. Takashima (1996). The structural analysis of neural networks using multilinear functions, electronics. *Information Communication Engineers,* D-II J79-D-II(7), 1271 1279.
[2] S.I. Gallant (1988). Connectionist expert systems. *Communication of ACM,* 31(2), 152-169.
[3] G.G. Towell and J.W. Shavlik (1993). Extracting refined rules from knowledge-based neural networks. *Machine Learning,* 13, 71-101.
[4] L.M. Fu (1994). Rule generation from neural networks. *IEEE Transactions on Systems, Man and Cybernetics,* 24(6), 1114-1124.
[5] A.G. Ivakhnenko (1966). The group method of data handling- a rival of the method of stochastic approximation. *Soviet Automatic Control,* 13(3), 43-55.
[6] H.R. Madala and A.G. Ivakhnenko (1994). *Inductive Learning Algorithm for Complex Systems Modeling,* pp. 291-296. CRC Press.
[7] A.G. Ivakhnenko and Y.U. Koppa (1970). Regularization of decision functions in the group method of data handling. *Soviet Automatic Control,* 15(2), 28-37.
[8J http://www.ics.uci.edu/AI/ML/Machine-Learning.html.
[9] J.R. Quinlan (1993). *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, San Mateo, California. (Japanese edition: Koichi Furukawa, Toppan, pp. 33-40. 1995.)

BIOGRAPHIES

Koji Fujimoto was born in Mie Prefecture, Japan. He received his BS in Mathematics from Sophia University, in 1985, and his PhD in Computer Science from Tokyo University of Agriculture and Technology in 1999, Japan. He is a consultant of Mathematical Modeling in financial business area at Financial Engineering Group, Inc. His research interests include inductive learning theory, data mining, text mining and time series data analysis.

Sampei Nakabayashi was born in Hokkaido Prefecture, Japan. He graduated from Jiyu Gakuen College of Liberal Arts. He started to work for Nippon Univac Research Institute in 1971. After two years of working, he worked as a social science system analyst for Nomura Research Institute, Ltd (NRI). NRI send him to the University of

Pennsylvania where he received M.S. in Public Policy and completed the Doctoral Program in Decision Science for one semester in 1980.

He established Financial Engineering Group, Inc. in 1989. He was installed the President in 1993. His interests include OR, Mathematical Statistics, System Decision Science in applied mathematics and AT.