EXTRACTING RULES FROM TRAINED RBF NEURAL NETWORKS
LIKUMU IEGŪŠANA NO APMĀCĪTIEM RBF NEIRONU TĪKLIEM

PETER GRABUSTS

Rezekne Higher Educational Institution
Atbrivoshanas al. 90, Rezekne LV-4600, Latvia
Phone: + 371 46 23798, *e-mail*: peter@ru.lv

Abstract. *This paper describes a method of rule extraction from trained artificial neural networks. The statement of the problem is given. The aim of rule extraction procedure and suitable neural networks for rule extraction are outlined. The RULEX rule extraction algorithm is discussed that is based on the radial basis function (RBF) neural network. The extracted rules can help discover and analyze the rule set hidden in data sets. The paper contains an implementation example, which is shown through standalone IRIS data set.*

Keywords: *neural networks, rule extraction, RBF networks, RULEX algorithm.*

Introduction

Nowadays a considerable effort is made to „write" and „read" symbolic information into and from artificial neural networks (ANN) [1, 2]. The motivation is multifold. ANNs have shown a very good ability to represent „empirical knowledge", as the one contained in a set of examples, but the information is expressed in a „sub-symbolic" form - in the structure, weights and biases of a trained ANN, not directly readable for the human user. So, an ANN behaves nearly like a „black box", providing no explanation to justify its decisions taken in various instances. This forbids the usage of ANNs in „safety-critical" domains, which include the economic and financial applications, and makes it difficult to verify and debug software that includes ANN components. On the other hand, the extraction of the knowledge contained in an ANN allows the „portability" of the information to other systems, in both symbolic (AI) and sub-symbolic (ANN) forms.

A direct way of converting neural to symbolic knowledge is through rule extraction. This process provides a limited form of an explanation facility of how a neural network may classify any given input pattern. Rule extraction is a process that discovers the hyper plane positions of the input-to-hidden units and the hidden-to-output units of a neural network. These positions are then formulated as IF..THEN rules with the most important input unit labels acting as the rule antecedents. The discovery of the hyper plane positions can be found by a number of techniques that analyze the weights and biases of the neural network. Rule extraction can be carried out using a variety of neural network types such as multi-layer perceptions, Kohonen networks, radial basis functions (RBF) and recurrent networks.

Rule extraction from RBF neural networks

The nature of RBF networks [3] makes them a suitable solution for rule extraction process. It is possible to extract a series of IF. THEN rules that are able to state simply and accurately the knowledge contained in the neural network. The RBF network consists of the feed forward architecture with an input layer, a hidden layer of RBF "pattern" units and an output layer of linear units. The input layer simply transfers the input vector to the hidden units, which form a localized response to the input pattern. Learning is normally undertaken as a two-stage process. The first stage consists of an unsupervised process in which the RBF centers (hidden units) are positioned and the optimum field widths are determined in relation to the training samples. The second stage of learning involves the calculation of the hidden unit to output unit weights and is achieved quite easily through a simple matrix transformation. The radial basis functions in the hidden layer are implemented by kernel functions, which operate over a localized area of input

ISBN 9984 – 779 – 06 – 8

space. The effective range of the kernels is determined by the values allocated to the center and width of the radial basis function. The Gaussian function has a response characteristic determined by equation [3]:

$$Z_j(x) = \exp(-\frac{\| x - \mu \|^2}{\delta_j^2})$$

The response of the output unit is calculated as

$$y = \sum_{j=i}^{J} W_{ij} Z_j(x)$$

where: W - weight matrix, Z - hidden units activations, x - input vectors, μ- parameter vector, σ- width of receptive field.

Rule extraction may be viewed in one of two ways. First, it can be seen as a technique for determining how the neural network performs any given input to output mapping. Second, the rule extraction process may often produce rules that are more accurate than the original neural network. The local nature of each RBF hidden unit enables a simple translation into a single rule:

IF $Feature_1$ is TRUE AND IF $Feature_2$ is TRUE AND IF $Feature_n$ is TRUE

THEN $Class_x$

where a Feature is composed of upper and lower bounds calculated by the RBF center μ_n positions, RBF width σ and feature steepness S. The value of the steepness was discovered empirically to be about 0.6 and is related to the value of the width parameter. The values of μ and σ are determined by the RBF training algorithm [3]. The upper and lower bounds are calculated as follows:

$X_{lower} = \mu_i - \sigma_i + S$ and $X_{upper} = \mu_i + \sigma_i - S$

The rule extraction algorithm RULEX [4] can be seen below in Fig. 1:

Input:	Hidden weights μ (center positions) Gaussian radius spread σ Steepness S
Output:	One rule per hidden unit
Procedure:	Train RBF network on data set For each hidden unit: For each μ_i $X_{lower} = \mu_i - \sigma_i + S$ $X_{upper} = \mu_i + \sigma_i - S$ Build rule by: antecedent=[X_{lower}, X_{upper}] Join antecedents with AND Add class label Write rule

Fig. 1. **Rule extraction algorithm**

Application example of rule extraction technique

The experiment performed was intended as an implementation of the RULEX algorithm that would give an idea of rule extraction possibilities from neural networks. The main aim of the experiment was to extract rules and test their quality. The software program is written in MATLAB. The present paper is a continuation of the study presented in [5].

The experiment employed the well-known Fisher's IRIS data set [6]. As known, it contains three flower classes of 50 elements each: setosa, versicolor and virginica. Every flower has 4 attributes:

- ✓ SL - sepal length;

ISBN 9984 – 779 – 06 – 8

- ✓ SW - sepal width;
- ✓ PL - petal length;
- ✓ PW - petal width.

Table 1 shows data fragments of each class.

Table 1.

IRIS data fragment

Setosa			
SL	SW	PL	PW
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2
4.6	3.1	1.5	0.2
.....	.....	.....	.....

Versicolor			
SL	SW	PL	PW
7.0	3.2	4.7	1.4
6.4	3.2	4.5	1.5
6.9	3.1	4.9	1.5
5.5	2.3	4.0	1.3
.....	.....	.....	.....

Virginica			
SL	SW	PL	PW
6.3	3.3	6.0	2.5
5.8	2.7	5.1	1.9
7.1	3.0	5.9	2.1
6.3	2.9	5.6	1.8
.....	.....	.....	.....

Using the GhostMiner statistical package, data normalisation was performed and element distribution by class as well as different statistical indexes were obtained (see Fig. 2 and 3).

	sepal length	sepal width	petal length	petal width
Minimum	0	0	0	0
Average	14.56	8.76	15.9467	8.55333
Maximum	34	22	42	21
Variance	90.8789	52.3447	160.762	45.4434
Std	9.53304	7.23496	12.6792	6.74118
Missing val.	0	0	0	0

Fig. 2. **Statistical indexes of the normalised Fisher's IRIS database**

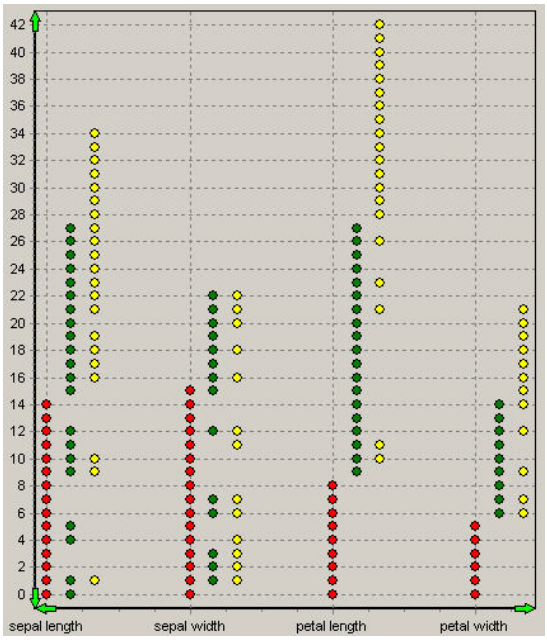

Fig. 3. **Distribution of elements by class**

For data visualisation, 2D projections can be used which show the distribution of particular parameters with regard to each other. Fig. 4.a illustrates the distribution of *petal length* with regard to *sepal length*, whereas Fig. 4.b depicts the distribution of *petal width* with regard to *sepal width*.

ISBN 9984 – 779 – 06 – 8

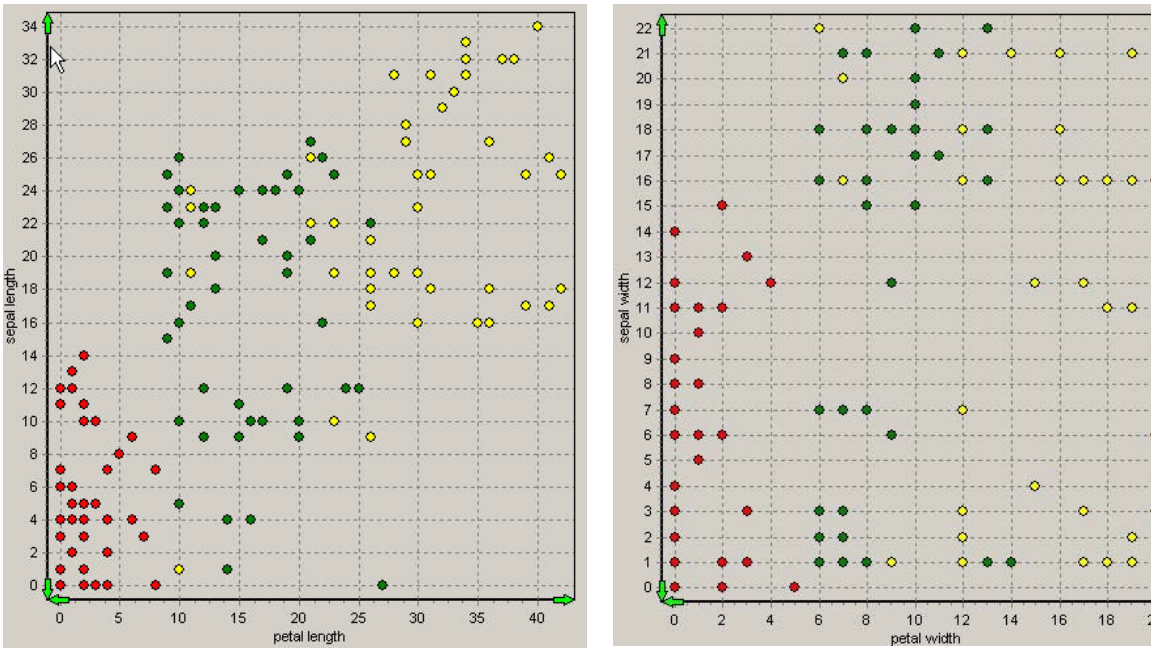

Fig. 4. **2D distributions of particular parameters: pl-sl (a) and pw-sw (b)**

The tasks of the experiment were as follows:

1. To accomplish network training by the RULEX algorithm at different training sets, namely:
   - Training set A - first 25 elements of every class;
   - Training set B - arbitrary 20 elements of every class;
   - Training set C - all 50 elements of every class.
2. To examine the effect of parameter S on the quality of extracted rules.

In case A, first 25 elements of every class were used as a training set. For all training sets, the values of parameter S were found experimentally. At each training stage, class centers and radius values were calculated according to the RULEX algorithm. Based on those values, for each class of training elements, X_{lower} and X_{upper} were calculated as well as the antecedent parts characterising the corresponding class. Then testing over the whole IRIS data set was made so as to determine to which extent the found rules correctly described elements of each class. For each class, the count of elements satisfying the rules was found as well as the percentage of elements correctly describing the rules. For convenience, IRIS variables SL, SW, PL and PW were denoted, respectively, as X1, X2, X3 and X4. Table 2 shows the results of the experiment, whereas Table 3 represents the rules extracted at different S values.

Table 2.

Results of training set A (first 25 elements of every class)

Correct	Values of parameter S													
	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.6
I	50	49	49	49	48	47	44	40	32	19	12	5	0	0
II	50	50	50	49	49	48	45	44	40	36	28	20	10	0
III	50	50	50	50	49	49	49	48	47	44	43	42	38	21
%	100	99.3	99.3	98.7	97.3	96	92	88	79.3	66	55.3	44.7	32	14

ISBN 9984 – 779 – 06 – 8

Table 3.

Training set A: characteristics of the extracted rules

	Parameter S=-0.9	Parameter S=0
Values of centers and radii	Class 1 = 5.03 3.48 1.46 0.25 Class 2 = 6.01 2.78 4.31 1.34 Class 3 = 6.58 2.93 5.64 2.04 Values of radii = 0.33 0.64 1.09	Class 1 = 5.03 3.48 1.46 0.25 Class 2 = 6.01 2.78 4.31 1.34 Class 3 = 6.58 2.93 5.64 2.04 Values of radii = 0.33 0.64 1.09
Percentage of rules correctly describing elements of classes	100	66

Rule of Class 1	IF (X1>= 3.80 AND < 6.26) AND
	IF (X2>= 2.25 AND < 4.71) AND
	IF (X3>= 0.23 AND < 2.69) AND
	IF (X4>= -0.98 AND < 1.48) THEN SETOSA
		IF (X1>= 4.70 AND < 5.36) AND
		IF (X2>= 3.15 AND < 3.81) AND
		IF (X3>= 1.13 AND < 1.79) AND
		IF (X4>= -0.08 AND < 0.58) THEN SETOSA

Rule of Class 2	IF (X1>= 4.47 AND < 7.55) AND
	IF (X2>= 1.24 AND < 4.31) AND
	IF (X3>= 2.77 AND < 5.85) AND
	IF (X4>= -0.19 AND < 2.88) THEN VERSICOLOR
		IF (X1>= 5.37 AND < 6.65) AND
		IF (X2>= 2.14 AND < 3.41) AND
		IF (X3>= 3.67 AND < 4.95) AND
		IF (X4>= 0.71 AND < 1.98) THEN VERSICOLOR

Rule of Class 3	IF (X1>= 4.58 AND < 8.57) AND
	IF (X2>= 0.94 AND < 4.92) AND
	IF (X3>= 3.65 AND < 7.63) AND
	IF (X4>= 0.05 AND < 4.04) THEN VIRGINICA
		IF (X1>= 5.48 AND < 7.67) AND
		IF (X2>= 1.84 AND < 4.02) AND
		IF (X3>= 4.55 AND < 6.73) AND
		IF (X4>= 0.95 AND < 3.14) THEN VIRGINICA

In case B, 20 elements, arbitrarily selected from every class, were employed as a training set. Table 4 demonstrates the results of the experiment, but Table 5 shows the rules extracted at different S values.

Table 4.

Results of training set B (arbitrary 20 elements of every class)

Correct	Values of parameter S													
	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.6
I	49	49	48	48	45	40	39	27	14	9	2	0	0	0
II	50	49	49	48	45	44	40	36	28	20	10	3	0	0
III	49	49	48	47	45	43	43	42	39	35	29	23	16	0
%	98.7	98	96.7	95.3	90	84.7	81.3	70	54	42.7	27.3	17.3	10.7	0

Table 5.

Training set B: characteristics of the extracted rules

	Parameter S=-0.9	Parameter S=0
Values of centers and radii	Class 1= 5.04 3.45 1.49 0.25 Class 2 = 5.99 2.77 4.32 1.35 Class 3 = 6.54 2.95 5.44 1.94 Values of radii = 0.19 0.43 0.73	Class 1 = 5.04 3.45 1.49 0.25 Class 2 = 5.99 2.77 4.32 1.35 Class 3 = 6.54 2.95 5.44 1.94 Values of radii= 0.19 0.43 0.73
Rules correctly describe elements of classes (%)	98.67	42.67

Rule of Class 1	IF (X1>= 3.95 AND < 6.13) AND
	IF (X2>= 2.36 AND < 4.54) AND
	IF (X3>= 0.40 AND < 2.59) AND
	IF (X4>= -0.84 AND < 1.34) THEN SETOSA
		IF (X1>= 4.85 AND < 5.23) AND
		IF (X2>= 3.26 AND < 3.64) AND
		IF (X3>= 1.30 AND < 1.69) AND
		IF (X4>= 0.06 AND < 0.44) THEN SETOSA

ISBN 9984 – 779 – 06 – 8

Rule of Class 2	IF (X1>= 4.66 AND < 7.32) AND IF (X2>= 1.44 AND < 4.10) AND IF (X3>= 2.99 AND < 5.65) AND IF (X4>= 0.01 AND < 2.68) THEN VERSICOLOR	IF (X1>= 5.56 AND < 6.42) AND IF (X2>= 2.34 AND < 3.20) AND IF (X3>= 3.89 AND < 4.75) AND IF (X4>= 0.91 AND < 1.78) THEN VERSICOLOR
Rule of Class 3	IF (X1>= 4.91 AND < 8.17) AND IF (X2>= 1.33 AND < 4.58) AND IF (X3>= 3.81 AND < 7.06) AND IF (X4>= 0.31 AND < 3.57) THEN VIRGINICA	IF (X1>= 5.81 AND < 7.27) AND IF (X2>= 2.23 AND < 3.68) AND IF (X3>= 4.71 AND < 6.16) AND IF (X4>= 1.21 AND < 2.67) THEN VIRGINICA

In case C, all 50 elements of every class served as a training set. Table 6 shows the results of the experiment, but Table 7 represents the rules extracted at different values of parameter S.

Table 6.

Results of training set C (all 50 elements of every class)

Correct	Values of parameter S													
	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.6
I	50	49	49	48	48	45	40	40	27	15	10	4	0	0
II	50	50	50	49	49	47	44	42	37	32	25	16	9	0
III	50	49	49	49	49	47	47	44	42	41	36	32	26	6
%	100	98.7	98.7	97.3	97.3	92.7	87.3	84	70.7	58.7	47.3	34.7	23.3	4

Table 7.

Training set C: characteristics of the extracted rules

	Parameter S=-0.9	Parameter S=0
Values of centers and radii	Class 1 = 5.01 3.42 1.46 0.24 Class 2 = 5.94 2.77 4.26 1.33 Class 3 = 6.59 2.97 5.55 2.03 Values of radii = 0.30 0.61 0.87	Class 1 = 5.01 3.42 1.46 0.24 Class 2 = 5.94 2.77 4.26 1.33 Class 3 = 6.59 2.97 5.55 2.03 Values of radii = 0.30 0.61 0.87
Rules correctly describe elements of classes (%)	100	58.7
Rule of Class 1	IF (X1>= 3.80 AND < 6.21) AND IF (X2>= 2.21 AND < 4.62) AND IF (X3>= 0.26 AND < 2.67) AND IF (X4>= -0.96 AND < 1.45) THEN SETOSA	IF (X1>= 4.70 AND < 5.31) AND IF (X2>= 3.11 AND < 3.72) AND IF (X3>= 1.16 AND < 1.77) AND IF (X4>= -0.06 AND < 0.55) THEN SETOSA
Rule of Class 2	IF (X1>= 4.42 AND < 7.45) AND IF (X2>= 1.26 AND < 4.28) AND IF (X3>= 2.75 AND < 5.77) AND IF (X4>= -0.19 AND < 2.84) THEN VERSICOLOR	IF (X1>= 5.32 AND < 6.55) AND IF (X2>= 2.16 AND < 3.38) AND IF (X3>= 3.65 AND < 4.87) AND IF (X4>= 0.71 AND < 1.94) THEN VERSICOLOR
Rule of Class 3	IF (X1>= 4.82 AND < 8.36) AND IF (X2>= 1.20 AND < 4.74) AND IF (X3>= 3.78 AND < 7.32) AND IF (X4>= 0.26 AND < 3.80) THEN VIRGINICA	IF (X1>= 5.72 AND < 7.46) AND IF (X2>= 2.10 AND < 3.84) AND IF (X3>= 4.68 AND < 6.42) AND IF (X4>= 1.16 AND < 2.90) THEN VIRGINICA

The data obtained prove that parameter S plays an essential role in the application of the RULEX algorithm: the greater the negative value of S is, the more the lower boundary of rule performance range, X_{lower}, decreases at the same time raising the upper boundary, X_{upper}, of the range. That causes the enlargement of the cluster describing antecedent part and thus increases

ISBN 9984 – 779 – 06 – 8

the value of the area in which the extracted rule is fulfilled. For training sets A, B, and C, the dependence of the total count of elements, correctly describing rules, on parameter S is shown in Table 8 and represented as a graph in Fig. 5.

Table 8.

The dependence of the total count (%) of rule satisfying elements on S

%	Values of parameter S													
	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.6
A	100	99.3	99.3	98.7	97.3	96	92	88	79.3	66	55.3	44.7	32	14
B	98.7	98	96.7	95.3	90	84.7	81.3	70	54	42.7	27.3	17.3	10.7	0
C	100	98.7	98.7	97.3	97.3	92.7	87.3	84	70.7	58.7	47.3	34.7	23.3	4

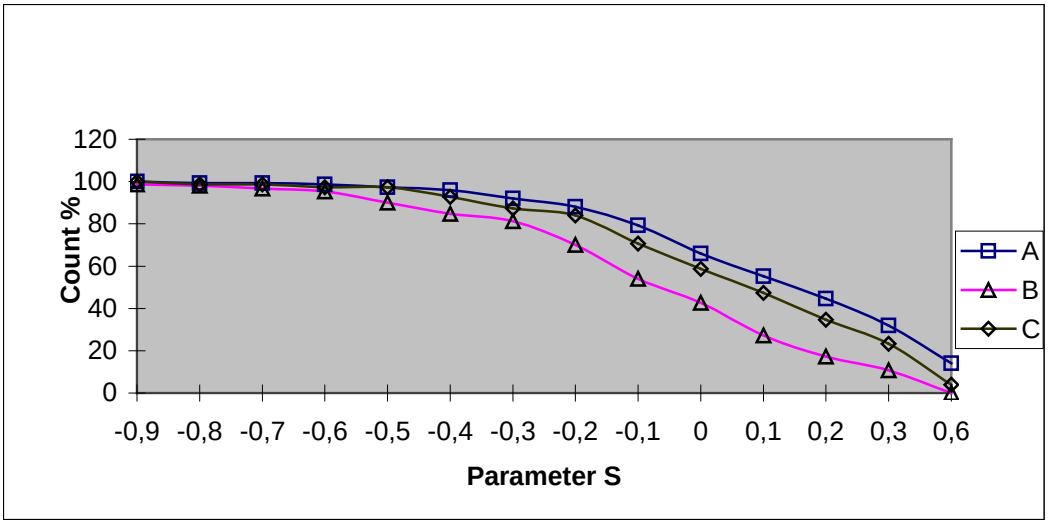

Fig. 5. **Dependence of the count (%) of elements correctly describing rules on parameter S**

Conclusions

The aim of this study was to continue the experiments described in [5]. In this paper a rule-extraction algorithm is shown which is based on the radial basis function (RBF) neural network classifier. After training the RBF classifier, the rules will be extracted through analyzing the parameters of the classifier. One hidden unit corresponds to one rule. Before extracting rules, the weights connecting the hidden units with output units are simplified. Then the interval for each input in the condition part of every rule is adjusted with a view to obtaining high rule accuracy by iteration steps. This rule extraction technique is shown through IRIS data set experimental results.

The extracted rules can help discover and analyse the hidden knowledge in data sets further.

References

1. Andrews, R., Diederich, J., Tickle, A. A survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-Based Systems, 8(6), 1995, p.373-389.
2. Crawen, M., Shavlik, J. Using sampling and queries to extract rules from trained neural networks. Machine Learning: Proceedings of the Eleventh International Conference, 1994.
3. Hush, D.R., Horne, B.G. Progress in Supervised Neural Networks. What's new since Lippmann? IEEE Signal Processing Magazine, vol.10, No 1, January 1993.
4. Andrews, R., Gewa, S. RULEX and CEBP networks as the basis for a rule refinement system. In: J. Hallam et al, editor, Hybrid Problems, Hybrid Solutions. IOS Press, 1995.
5. Grabusts P. Neural networks methods of knowledge extraction. Proceedings of the International Conference "Scientific Achievements for Wellbeing and Development of Society", March 4-5, Rezekne, Rezekne Higher Educational Institution, 2004, p. 99-106.
6. Fisher R.A. The use of multiple measurements in taxonomic problems. Ann. Eugenics, 7(2), 1936, p.179-188.

ISBN 9984 – 779 – 06 – 8