Two ways of using artificial neural networks in knowledge discovery from chemical materials data*

Martin Holeňa

Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2, 18207 Prague
martin@cs.cas.cz, web:cs.cas.cz/~martin

Abstract. *In the application area of chemical materials, data mining methods have been used for more than a decade. By far most popular have from the very beginning been methods based on artificial neural networks. However, they are frequently used without awareness of the difference between the numeric nature of knowledge obtained from data by neural network regression, and the symbolic nature of knowledge obtained by some other data mining methods. This paper explains that within the surrogate modelling approach, which plays an important role in this area, using numeric knowledge is justified. At the same time, it recalls the possibility to obtain symbolic knowledge from neural networks in the form of logical rules and describes a recently proposed method for the extraction of Boolean rules in disjunctive normal form. Both ways of using neural networks are illustrated on examples from this application area.*

1 Introduction

The search for new chemical materials, e.g., catalytic materials for a plethora of chemical reactions, produces large amounts of data. To discover useful knowledge from those data, statistical as well as machine-learning data mining methods have been used in this area since the late 1990s, the former represented in particular by the analysis of variance, decision trees and support vector regression, the latter by main variants of feed-forward neural networks.

This paper summarizes experience from nearly ten years using and developing neural-networks based data mining methods for catalytic data. Artificial neural networks are the most popular regression model in this application area. In the survey [10], more than 20 published applications of multilayer perceptrons (MLPs) to catalytic data have been listed, as well as several applications of radial basis function networks. The role of feed-forward neural nets as a regression model predicting catalytic performance of materials (such as yield, conversion, selectivity) is due partially to their preceding success in other areas, but mostly to their ability to serve as universal approximators in very general function spaces [12, 14, 18]. This ability is particularly valuable in the context of the highly nonlinear nature of the dependencies encountered in catalysis (cf. Figure 1).

However, it seems to be little awareness, among researchers using artificial neural networks in catalysis, of the difference among the symbolic nature of the knowledge obtained from data by analysis of variance and decision trees, and the numeric nature of the knowledge obtained by neural network regression.

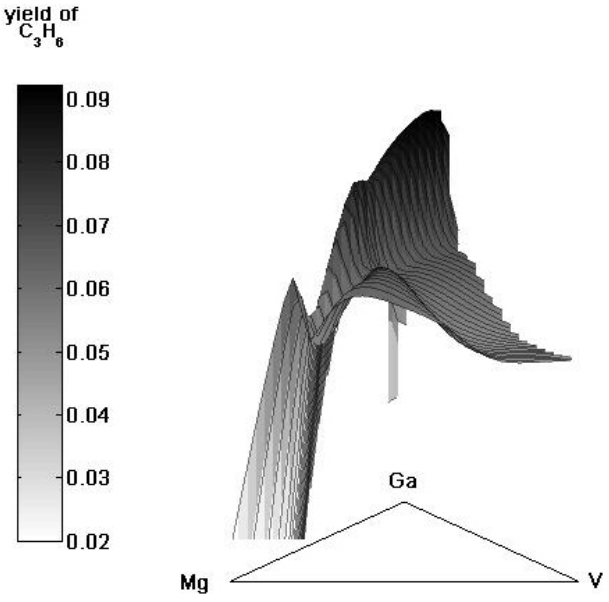

Fig. 1. A 3-dimensional cut of a neural-network regression of the yield of a reaction product on the composition of the catalytic material.

Incited by the situation just outlined, the paper presents two strategies for the application of artificial neural networks to data about chemical materials. The first strategy relies on the numeric knowledge from neural network regression. Although numeric knowledge is much less understandable to humans than symbolic knowledge (in terms of [4], it has a high "data fit", but a low "mental fit"), we show that in this application area, it can be very useful if directly integrated

* The research reported in this paper has been supported by the grant No. 201/08/1744 of the Grant Agency of the Czech Republic and partially supported by the Institutional Research Plan AV0Z10300504.

with the optimization of materials performance in an approach called surrogate modelling. In that context, also the possibility to increase the accuracy of neural network regression by means of boosting is mentioned. The other strategy, on the other hand, relies on employing rules extraction methods to obtain, from trained neural networks, symbolic knowledge.

These two strategies determine also the structure of the paper. In Section 2, the surrogate modelling approach is described. Section 3 then explains a method for the extraction of logical rules from trained neural networks. Both strategies are in the respective sections illustrated using real-world examples.

2 Neural networks used as surrogate models

From the point of view of theoretical computer science, the search for most suitable chemical materials entails complex optimization tasks. As objective functions, those tasks use various properties of the materials, e.g. in the case of catalytic materials, properties quantifying their catalytic performance, such as yield, conversion, or selectivity. A crucial feature of such objective functions is that they cannot be expressed analytically, their values must be obtained empirically. For their optimization, it is not possible to employ most common optimization methods, such as steepest descent, conjugate gradient methods or the Levenberg-Marquardt method. Indeed, to obtain sufficiently precise numerical estimates of gradients or second order derivatives of the empirical objective function, those methods need to evaluate the function in points some of which would have a smaller distance than is the empirical error of catalytic measurements. That is why methods not requiring any derivatives have been used to solve such optimization tasks, such as the simplex method, and most frequently genetic and other evolutionary algorithms [2]. To compensate for missing information about derivatives, these methods need quite large number of objective function evaluations. In the context of catalysis, this is quite disadvantageous because the evaluation of the empirical objective functions used in the search for optimal catalysts is often costly and time-consuming. Testing a generation of catalytic materials proposed by an evolutionary algorithm typically needs several days of time and costs thousands of euros.

The usual approach to decreasing the cost and time of optimization of empirical objective functions is to evaluate the function only in points considered to be most important for the progress of the employed optimization method, and to evaluate its suitable regression model otherwise. That model is termed *surrogate model* of the function, and the approach is referred to as *surrogate modelling* [17, 20, 23, 27]. Needless to say, the time and costs needed to evaluate a regression model are negligible compared to time and costs needed to evaluate empirical functions such as yield or conversion. However, it must not be forgotten that the agreement between the results obtained with a surrogate model and those obtained with the original function depends on the accuracy of the model.

The fact that feed-forward neural networks are the most frequent regression models in catalysis suggests them as the most natural candidate for surrogate models in this area. Indeed, several nice examples of the application of neural-network based surrogate modelling to the optimization of performance of catalytic materials have been published during the last five years [3, 6, 21, 25]. Within the overall context of the application of artificial neural networks to mining catalytic data, however, they are still rare.

Although surrogate modelling has been also applied to conventional optimization [5], it is most frequently encountered in connection with evolutionary algorithms because for them, the approach leads to the approximation of the fitness function, whose usefulness in evolutionary computation is already known [13, 19]. For the progress of evolutionary optimization, most important criteria are on the one hand points that indicate closeness to the global optimum (through highest values of the fitness function), on the other hand points that most contribute to the diversity of the population.

In the literature, various possibilities of combining evolutionary optimization with surrogate modelling have been discussed [17, 24, 27]. Nevertheless, all of them are controlled by one of two basic approaches:

A. The *individual-based-control* consists in choosing between the evaluation of the empirical objective function and the evaluation of its surrogate model individual-wise, basically in the following steps:
   - **(i)** An initial set $\mathcal{E}$ of individuals is collected, in which the considered empirical fitness η was evaluated (for example, the population of several first generations of the evolutionary algorithm).
   - **(ii)** The surrogate model is constructed using the set of pairs $\{(x, \eta(x)) : x \in \mathcal{E}\}$.
   - **(iii)** The evolutionary algorithm is run with the fitness η replaced by the model for one generation with a population $\mathcal{Q}$ of size qP, where P is the desired population size for the optimization of η, and q is a prescribed ratio (e.g., $q = 10$ or $q = 100$).
   - **(iv)** A subset $\mathcal{P} \subset \mathcal{Q}$ of size P is selected so as to contain those individuals from $\mathcal{Q}$ that are most important according to the considered criteria for the progress of optimization.

(v) For $x \in \mathcal{P}$, the empirical fitness is evaluated.

(vi) The set $\mathcal{E}$ is replaced by $\mathcal{E} \cup \mathcal{P}$ and the algorithm returns to the step (ii).

B. The *generation-based-control* consists in choosing between both kinds of evaluation generation-wise, basically in the following steps:

(i) An initial set $\mathcal{E}$ of individuals in which the considered empirical fitness η was evaluated is collected like with the individual-based control.

(ii) The surrogate model is constructed using the set of pairs $\{(x, \eta(x)) : x \in \mathcal{E}\}$.

(iii) Relying on the error of the surrogate model, measured with a prescribed error measure (e.g., *mean squared error*, MSE, or *mean absolute error*, MAE), an appropriate number g_m of generations is chosen, during which η should be replaced by the model.

(iv) The evolutionary algorithm is run with the fitness η replaced by the model for g_m generations with populations $\mathcal{P}_1, \ldots, \mathcal{P}_{g_m}$ of size P.

(v) The evolutionary algorithm is run with the empirical fitness η for a prescribed number g_e of generations (frequently, $g_e = 1$) with populations $\mathcal{P}_{g_m+1}, \ldots, \mathcal{P}_{g_m+g_e}$.

(vi) The set $\mathcal{E}$ is replaced by $\mathcal{E} \cup \mathcal{P}_{g_m+1} \cup \ldots \cdots \cup \mathcal{P}_{g_m+g_e}$ and the algorithm returns to the step (ii).

The agreement between the results that are obtained with a surrogate model and those that would be obtained if the empirical objective function were evaluated depends on the accuracy of the model. A popular approach to increasing the accuracy of learning methods is boosting, i.e., construction of a strong learner through combining weak learners. It is important to realize that boosted surrogate models are only particular kinds of surrogate models and their interaction with optimization algorithms in optimization tasks follows the same rules as the interaction of surrogate models in general. In particular in the above outlines of individual-based and generation-based control, boosting is always performed in the step (ii), which has to be replaced with:

(ii'a) The set $\{(x, \eta(x)) : x \in \mathcal{E}\}$ is divided into k disjoint subsets of size $\lfloor \frac{|\mathcal{E}|}{k} \rfloor$ or $\lceil \frac{|\mathcal{E}|}{k} \rceil$, where $|\ |$ denotes the cardinality of a set, $\lfloor\ \rfloor$ the lower integer bound of a real number, and $\lceil\ \rceil$ its upper integer bound.

(ii'b) For each $j = 1, \ldots, k$, a surrogate model F_1^j is constructed, using only data not belonging to the j-th subset.

(ii'c) A k-fold crossvalidation of regression boosting is performed, and the error of the boosting approximation is in each iteration measured with the prescribed error measure on the validation data.

(ii'd) The first iteration i in which the average error of the boosting approximation on the validation data is lower than in the $i+1$-th iteration is taken as the final iteration of boosting.

(ii'e) Boosting using the complete set $\{(x, \eta(x)) : x \in \mathcal{E}\}$ is performed up to the final iteration found in step (ii'd), and the result of the application of the employed boosting method in each such iteration of boosting is taken as the boosted surrogate model in that iteration.

2.1 An illustration

A particular method for MLP boosting has been presented in [11]. That method will now be employed in surrogate modelling with data from the investigation of catalytic materials for the high-temperature synthesis of hydrocyanic acid (HCN) [16]. The composition of most of those materials was designed by means of a specific genetic algorithm (GA) for heterogeneous catalysis [26]. As usually in evolutionary optimization of catalytic materials, the GA configuration was determined by the experimental conditions in which the optimization was performed: number of channels of the reactor in which the materials were tested, as well as time and financial resources available for those expensive tests. In the reported investigation, the algorithm was running for 7 generations of population size 92, and in addition 52 other catalysts with manually designed composition were investigated. Consequently, data about 696 catalytic materials were available. The considered MLPs had *14 input neurons*: 4 of them coding catalyst support, the other 10 corresponding to the proportions of 10 metal additives forming the active shell, and *3 output neurons*, corresponding to 3 kinds of catalytic activity considered as fitness functions.

For boosting, only data about catalysts from the 1.-6. generation of the GA and about the 52 catalysts with manually designed composition were used, thus altogether data about 604 catalytic materials. Data about catalysts from the 7. generation were completely excluded and left out for testing. The set of architectures to which boosting was applied was restricted to MLPS with 1 and 2 hidden layers and was delimited by means of the heuristic pyramidal condition: the number of neurons in a subsequent layer must not exceed the number of neurons in a previous layer. Let n_I, n_H and n_O denote the numbers of input, hidden and output neurons, respectively, and n_{H1} and n_{H2} denote the numbers of neurons in the first and second hid-

den layer, respectively. Then the pyramidal condition entails the following 90 architectures:

(i) one hidden layer and $3 \leq n_H \leq 14$ (12 architectures);

(ii) two hidden layers and $3 \leq n_{H2} \leq n_{H1} \leq 14$ (78 architectures).

As was mentioned above, boosting can be combined both with the individaul-based and with the generation-based control of surrogate modelling. In the reported investigation of catalytic materials for HCN synthesis, the indiviual-based control was employed.

The error measure employed in the crossvalidation in the step (ii'c) was MSE. The distribution of the final iterations of boosting, found for MLPs with the 90 considered architectures in the step (ii'd), is depicted in Figure 2. We can see that only for 16 MLPs, already the 1st iteration was the final. For the remaining 74 MLPs, boosting improved the average MSE on the validation data for at least 1 iteration. The mean and median of the distribution of the final iterations were 6.6 and 5, respectively.

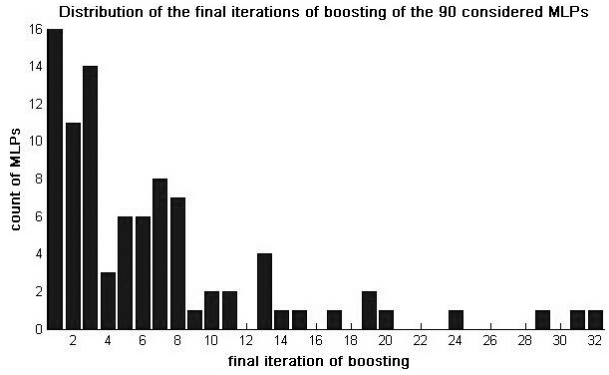

Fig. 2. Distribution of the final iterations of boosting of the 90 MLPs with 1-hidden-layer architectures fulfilling $3 \leq n_H \leq 14$ and 2-hidden-layer architectures fulfilling $3 \leq n_{H2} \leq n_{H1} \leq 14$.

For testing with the data from the 7th generation of the evolutionary algorithm, we used only the five MLPs most promising from the point of view of the average MSE on the validation data in the final iteration of boosting. These were the following MLPs:

- a 1-hidden-layer MLP, with $n_H = 11$ and the 3rd iteration of boosting being the final iteration,
- four 2-hidden-layers MLPs, with $(n_{H1}, n_{H2}) = (10,4), (10,6), (13,5), (14,8)$ and the final iterations of boosting 19, 32, 31 and 29, respectively.

For each of them, the validation proceeded as follows:

1. In each iteration up to the final, a single MLP was trained with data about all the 604 catalytic materials used for boosting.
2. In each iteration up the final iteration of boosting, the boosted surrogate model was constructed for the trained MLP, according to the step (ii'e).
3. From the values predicted by the boosted surrogate model for the 92 materials from the 7. generation of the GA, and from the measured values, the boosting MSE was calculated.

The results are summarized in Figure 3, decomposed to the properties corresponding to the MLP outputs – conversions of CH_4 and NH_3 and yield of HCN. They clearly confirm the usefulness of boosting for the five considered architectures. For each of them, boosting leads to an overall decrease of MSE of the conversion of CH_4 and HCN yield, on new data from the 7th generation of the GA, which is uninterrupted or nearly uninterrupted till the final boosting iteration. On the other hand, boosting did not lead to any decrease of the error of the conversion of NH3, which on the other hand is already from the beginning much lower than the two other performance measures (notice that the scale of the y-axis is 10-times finer for the conversion of NH_3 than for the conversion of CH_4 and HCN yield). The explanation for the different behavior of the conversion of NH_3 is the substantially lower variability of its values in the seventh generation of the GA, used for validating the usefulness of boosting (standard deviation, SD: 2.8, interquartile range, IQR: 1.6), compared to the conversion of CH_4 (SD: 26.1, IQR: 45.0) and HCN yield (SD: 20.1, IQR: 35.9). Due to so low variability, the conversion of NH_3 appears effectively as nearly constant during the validation of boosting, which in turn accounts for a nearly constant MSE.

3 Neural-network based rules extraction from data

The architecture of a trained neural network and the weights and biases that determine the regression model computed by the network inherently represent the knowledge contained in the data used to train the network. As was already mentioned in the introduction, such a representation is not comprehensible to humans, being very far from the symbolic, modular and often vague way they represent knowledge by themselves. Therefore, methods for the extraction of symbolic knowledge from trained neural networks have been investigated since the late 1980s. Most frequently, the extracted knowledge has the form of a Boolean implication:

IF the input variables fulfil an input condition C_I

THEN the output variables are likely
to fulfil an output condition C_O. (1)

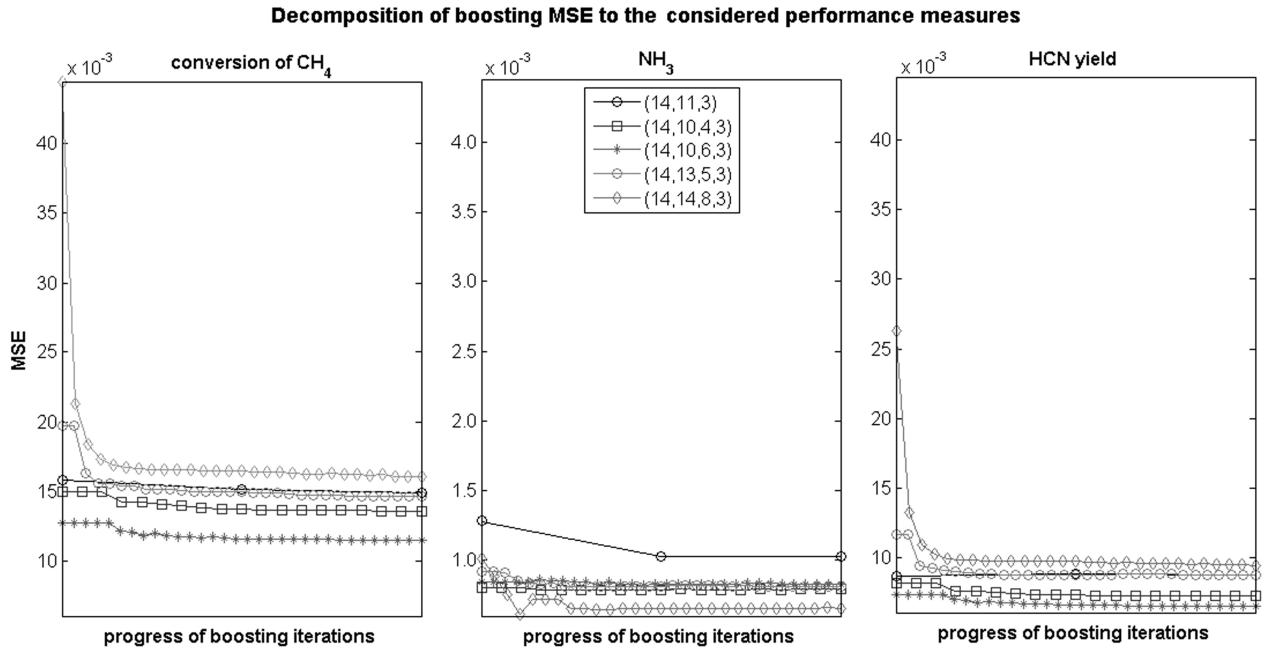

Fig. 3. History of the boosting MSE on the data from the 7th generation of the GA for MLPs with the 5 architectures included in the validation of boosting, decomposed to the properties corresponding to the MLP outputs.

In addition, also implications and equivalences of important kinds of fuzzy logic are frequently extracted [8, 15]. In general, extracted formulas of a formal logic are called *rules*. Over the last two decades, various rules extraction methods have been proposed for neural networks, but so far none of them has become a common standard (cf. the survey papers [1, 15, 22] and the monograph [7]). Here, a method for the extraction of Boolean implications from multilayer perceptrons with n inputs and m outputs will be sketched that finds to each output condition of the form:

$$C_O: \text{ the value } y \text{ of the output variables lies in a rectangular area } R \subset \mathcal{R}^m \quad (2)$$

one or more input conditions of the form

$$C_I: \text{ the value } x \text{ of the input variables lies in a polyhedron } P \subset \mathcal{R}^n \quad (3)$$

Hence, this method extracts rules of the form:

$$\text{IF } x \in P \text{ THEN } y \in R. \quad (4)$$

A detailed explanation of the method can be found in [9]. Its main principles can be summarized as follows:

- An m-dimensional rectangular area R with borders perpendicular to the m coordinate axes has to be chosen in advance in the output space of a trained MLP with sigmoid activation functions. The reason for choosing such an area is that in the space of evaluations of m free variables, each m-dimensional rectangular area is the validity set of the conjunction of some m univariate Boolean predicates. That conjunction then serves as the consequent of the rule to extract.
- The activation functions in the hidden neurons are approximated with piecewise-linear sigmoid activation functions. This can be done with an arbitrary precision.
- The products of individual linearity intervals of all the activation functions determine areas in the input space in which the final approximating mapping computed by the multilayer perceptron is linear.
- In each such area, all points mapped to R form a polyhedron, which may eventually be empty or may be concatenated with polyhedra from some of the neighboring areas to a larger polyhedron.
- The union of all the nonempty concatenated polyhedra $P_1, \ldots, P_q$ defines the antecedent of a rule in a combined form

$$\text{IF } x \in P_1 \cup \cdots \cup P_q \text{ THEN } y \in R, \quad (5)$$

which is equivalent to a logical disjunction of q rules of the simple form (4):

$$\begin{aligned} &\text{IF } x \in P_1 \text{ THEN } y \in R \\ &\quad \dots \\ &\text{IF } x \in P_q \text{ THEN } y \in R. \end{aligned} \tag{6}$$

To increase the comprehensibility of the extracted rules, visualization by means of 2- or 3-dimensional cuts of the set $P_1 \cup \dots \cup P_q$ can be used (Figure 4).

Usually, logical rules of the form (4) are the final results of this rule-extraction method. Nonetheless, there is one exception – when the polyhedron P is also rectangular with borders perpendicular to axes, or more generally, when P can be approximately replaced with such a rectangular area R_I in the input space. Then the above rule (4) can be approximately expressed in the conjunctive form

$$\text{IF } x_1 \in I_1 \,\&\, \dots \,\&\, x_{n_I} \in I_{n_I} \text{ THEN } y \in R. \tag{7}$$

Here, $I_1, \dots, I_{n_I}$ are intervals that constitute the projections of R_I into the n_I input dimensions. Each such interval can be restricted both from below and from above, restricted only from below or only from above, or finally can be even the complete set of real numbers. However, dimensions for which the corresponding projection of R_I equals the complete real axis are usually not included in (7), since they would not provide any new knowledge. Finally, observe that due to (5) and (7), the final extracted rule is in the *disjunctive normal form*.

In the rule-extraction method outlined above, the possibility of replacing a polyhedron P with a rectangular area R_I is assessed according to the following principles:

1. The resulting dissatisfaction with points that either belong to P but do not belong to R_I, or belong to R_I but do not belong to P (i.e., with points from the symmetric difference $R_I \Delta P$), has to remain within a prescribed tolerance ε and R_I has to be minimal in the input space among rectangular areas of some specified kind with dissatisfacion within that tolerance.
2. The dissatisfaction with points from $R_I \Delta P$ depends solely on those points and is increasing with respect to inclusion. Consequently, it can be measured using some monotone measure on the input space, possibly depending on P.
3. To be eligible for replacement, P has to cover at least one point of the available data.

For 2., the most attractive monotone measures, due to their straightforward interpretability, are:

– The joint empirical distribution of the input variables in the available data.

– The conditional empirical distribution of the input variables in the available data, conditioned by P.

Rules of the form (7) are also very convenient from the visualization point of view: Since cuts of rectangular areas coincide with the corresponding projections of those areas, the values of no variables need to be fixed.

3.1 An illustration

As an example, Figure 5 shows three-dimensional cuts determining the antecedents of conjunctive-form rules extracted from a trained MLP with 5 input neurons and 1 output neuron such that:

(i) the input neurons correspond to variables that record the molar proportions of the oxides of Fe, Ga, Mg, Mn and Mo in the catalytic material;

(ii) the output neuron corresponds to a variable recording propene yield.

The extracted rules are listed in Table 1.

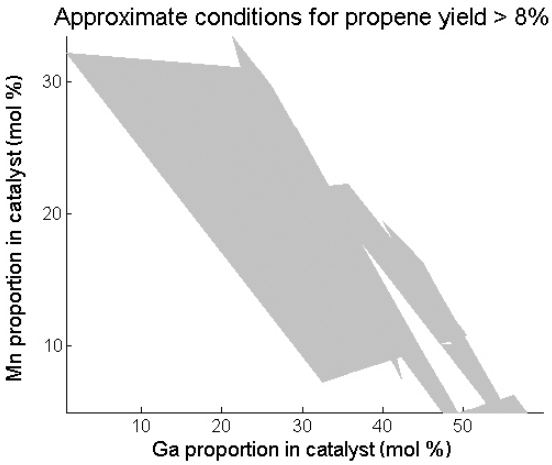

Fig. 4. A two-dimensional cut of the union of polyhedra from the antecedent of a rule of the form (5) extracted from a trained MLP. The cut corresponds to input variables recording the molar proportions of oxides of Mn and Ga in the catalytic material, for the consequent "propene yield $> 8\%$".

4 Conclusion

The paper dealt with employing feed-forward neural networks for knowledge discovery from data about chemical materials. It has shown that in this application area, obtaining numeric knowledge by neural-network regression is justified, in spite of the fact that numeric knowledge is substantially less human-understandable

Rule	Antecedent	Consequent
1	24% < Ga proportion < 33% & 31% < Mg proportion < 39% & Mo proportion < 7% & Fe, Mn proportions = 0	C_3H_6 yield > 8%
2	Ga proportion ≈ 36% & 28% < Mg proportion < 38% & Fe, Mn, Mo proportions = 0	
3	Fe proportion < 12% & Ga proportion ≈ 38% & 29% < Mg proportion < 36% & Mo proportion < 9% & Mn proportion = 0	

Table 1. Antecedents of the rules of the form (7) extracted using the method described in this section for the consequent "propene yield > 8%" from a trained MLP with 5 input neurons and 1 output neuron, assuming that the above interpretation of the variables to which those neurons correspond is described by (i) and (ii).

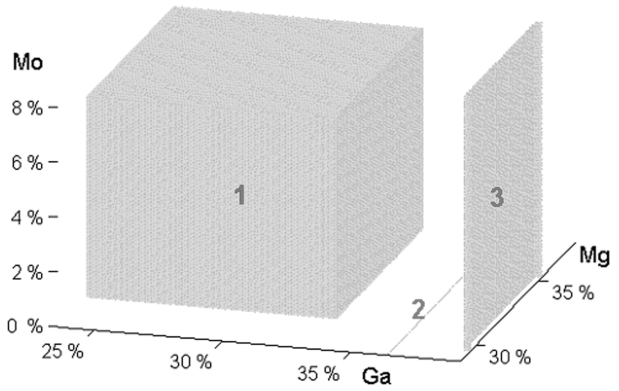

Fig. 5. A three-dimensional projection of the union of rectangular areas that replace, following the method described in this section, the union of of polyhedra from the antecedent of a combined form rule extracted from a trained MLP. The projection corresponds to input variables recording the molar proportion of oxides of Ga, Mg and Mo in a catalytic material. The numbers 1, 2, 3 refer to the antecedents of the rules in Table 1.

than symbolic knowledge. Its justification consists in the possibility to use such knowledge in the optimization tasks entailed by search for new materials in the surrogate modelling approach.

In addition to justifying the specific need for numeric knowledge from neural network regression in this application area, the paper recalled the possibility to obtain symbolic knowledge in the form of logical rules from trained neural networks. It explained a recently proposed method for the extraction of Boolean rules in disjunctive normal form, and illustrated it on data about catalytic materials.

References

1. R. Andrews, J. Diederich, and A.B. Tickle: *Survey and critique of techniques for extracting rules from trained artificical neural networks.* Knowledge Based Systems, **8**, 1995, 378–389.
2. M. Baerns and M. Holeňa: *Combinatorial development of solid catalytic materials. Design of high-throughput experiments, data analysis, data mining.* World Scientific, Singapore, 2009.
3. L.A. Baumes, D. Farrusseng, M. Lengliz, and C. Mirodatos: *Using artificial neural networks to boost high-throughput discovery in heterogeneous catalysis.* QSAR and Combinatorial Science, **23**, 2004, 767–778.
4. M. Berthold and D. Hand (eds): *Intelligent data analysis. An introduction.* Springer Verlag, Berlin, 1999.
5. A.J. Brooker, J. Dennis, P.D. Frank, D.B. Serafini, Torczon V., and M. Trosset: *A rigorous framework for optimization by surrogates.* Structural and Multidisciplinary Optimization, **17**, 1998, 1–13.
6. D. Farrusseng, F. Clerc, C. Mirodatos, N. Azam, F. Gilardoni, J.W. Thybaut, P. Balasubramaniam, and G.B. Marin: *Development of an integrated informatics toolbox: HT kinetic and virtual screening.* Combinatorial Chemistry and High Throughput Screening, **10**, 2007, 85–97.
7. A.S.A. Garcez, L.C. Lamb, and D.M. Gabbay: *Neural-symbolic cognitive reasoning.* Springer Verlag, Berlin, 2009.
8. M. Holeňa: *Extraction of fuzzy logic rules from data by means of artificial neural networks.* Kybernetika, **41**, 2005, 297–314.
9. M. Holeňa: *Piecewise-linear neural networks and their relationship to rule extraction from data.* Neural Computation, **18**, 2006, 2813–2853.
10. M. Holeňa and M. Baerns: *Computer-aided strategies for catalyst development.* In: G. Ertl, H. Knözinger, F. Schüth, and J. Eitkamp, (eds), Handbook of Heterogeneous Catalysis, Wiley-VCH, Weinheim, 2008, 66-81.
11. M. Holeňa, D. Linke, and N. Steinfeldt: *Boosted neural networks in evolutionary computation.* In: Neural Information Processing. Lecture Notes in Computer Science 5864, Springer Verlag, Berlin, 2009, 131–140.
12. K. Hornik: *Approximation capabilities of multilayer neural networks.* Neural Networks, **4**, 1991, 251–257.
13. Y. Jin, M. Hüsken, M. Olhofer, and B. Sendhoff: *Neural networks for fitness approximation in evolutionary optimization.* In Y. Jin, (ed.), Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 281–306.

14. V. Kůrková: *Neural networks as universal approximators.* In: M. Arbib, (ed.), Handbook of Brain Theory and Neural Networks, MIT Press, Cambridge, 2002, 1180–1183.
15. S. Mitra and Y. Hayashi: *Neuro-fuzzy rule generation: Survey in soft computing framework.* IEEE Transactions on Neural Networks, **11**, 2000, 748–768.
16. S. Möhmel, N. Steinfeldt, S. Endgelschalt, M. Holeňa, S. Kolf, U. Dingerdissen, D. Wolf, R. Weber, and M. Bewersdorf: *New catalytic materials for the high-temperature synthesis of hydrocyanic acid from methane and ammonia by high-throughput approach.* Applied Catalysis A: General, **334**, 2008, 73–83.
17. Y.S. Ong, P.B. Nair, A.J. Keane, and K.W. Wong: *Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems.* In: Y. Jin, (ed.), Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 307-331.
18. A. Pinkus: *Approximation theory of the MPL model in neural networks.* Acta Numerica, **8**, 1998, 277–283.
19. A. Ratle: *Accelerating the convergence of evolutionary algorithms by fitness landscape approximation.* In: A.E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, (eds), Parallel Problem Solving from Nature, Springer Verlag, Berlin, 1998, 87–96.
20. A. Ratle: *Kriging as a surrogate fitness landscape in evolutionary optimization.* Artificial Intelligence for Engineering Design, Analysis and Manufacturing, **15**, 2001, 37–49.
21. U. Rodemerck, M. Baerns, and M. Holeňa: *Application of a genetic algorithm and a neural network for the discovery and optimization of new solid catalytic materials.* Applied Surface Science, **223**, 2004, 168-174.
22. A.B. Tickle, R. Andrews, M. Golea, and J. Diederich: *The truth will come to light: Directions and challenges in extracting rules from trained artificial neural networks.* IEEE Transactions on Neural Networks, **9**, 1998, 1057–1068.
23. H. Ulmer, F. Streichert, and A. Zell: *Model-assisted steady state evolution strategies.* In: GECCO 2003: Genetic and Evolutionary Computation, Springer Verlag, Berlin, 2003, 610–621.
24. H. Ulmer, F. Streichert, and A. Zell: *Model assisted evolution strategies.* In: Y. Jin, (ed.), Knowledge Incorporation in Evolutionary Computation, Springer Verlag, Berlin, 2005, 333–355.
25. S. Valero, E. Argente, V. Botti, J.M. Serra, P. Serna, M. Moliner, and A. Corma: *DoE framework for catalyst development based on soft computing techniques.* Computers and Chemical Engineering, **33**, 2009, 225-238.
26. D. Wolf, O.V. Buyevskaya, and M. Baerns: *An evolutionary approach in the combinatorial selection and optimization of catalytic materials.* Applied Catalyst A: General, **200**, 2000, 63–77.
27. Z.Z. Zhou, Y.S. Ong, P.B. Nair, A.J. Keane, and K.Y. Lum: *Combining global and local surrogate models to accellerate evolutionary optimization.* IEEE Transactions on Systems, Man and Cybernetics. Part C: Applications and Reviews, **37**, 2007, 66–76.