Available online at www.sciencedirect.com
SCIENCE DIRECT®
Fuzzy Sets and Systems 149 (2005) 187–207
FUZZY sets and systems
www.elsevier.com/locate/fss

Knowledge discovery by a neuro-fuzzy modeling framework

G. Castellano*, C. Castiello, A.M. Fanelli, C. Mencar

Computer Science Department, University of Bari, Via Orabona, 4, Bari 70126, Italy

Available online 23 August 2004

Abstract

In this paper a neuro-fuzzy modeling framework is proposed, which is devoted to discover knowledge from data and represent it in the form of fuzzy rules. The core of the framework is a knowledge extraction procedure that is aimed to identify the structure and the parameters of a fuzzy rule base, through a two-phase learning of a neuro-fuzzy network. In order to obtain reliable and readable knowledge, two further stages are integrated with the knowledge extraction procedure: a pre-processing stage, performing variable selection on the available data to obtain simpler and more reliable fuzzy rules, and a post-processing stage, that granulates outputs of the extracted fuzzy rules so as to provide a validity range of estimated outputs. Moreover, the framework can address complex multi-input multi-output problems. In such case, two distinct modeling strategies can be followed with the opportunity of producing both a single MIMO model or a collection of MISO models. The proposed framework is verified on a real-world case study, involving prediction of chemical composition of ashes produced by combustion processes carried out in thermo-electric generators located in Italy.

Keywords: Knowledge extraction from data; Neuro-fuzzy systems; Feature selection; Fuzzy clustering; Learning

1. Introduction

The knowledge-data trade off represents a key issue in the field of modeling: both the available observed data and the initial prior knowledge can be exploited to construct models that mimic relationships among data. Modeling takes place between two extremes which in their "pure form" can be identified in the following ideal conditions: (i) the initial knowledge is nil and the model is extracted by a learning process

* Corresponding author. Dipartimento di Informatica, Universita degli Studi di Bari, Via E. Orabona 4, Bari 70126, Italy. Tel.: +39-080-544-32-85; fax: +39-080-544-31-96.
E-mail address: castellano@di.uniba.it (G. Castellano).

0165-0114/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.fss.2004.07.015

based on a certain amount of data; (ii) the initial knowledge is complete and correct and the model can be learnt with no need of any data. In the first case, inductive methods are applied to generalize the information deriving from a particular set of data to a wider context (*data-driven learning*). On the other hand, a deductive inference is performed when the general purpose a priori knowledge is employed to reformulate what is already known in order to increase efficiency in the performance of certain tasks (*knowledge driven learning* or *analytical learning*) [15,33].

As known, different approaches have been proposed to develop artificial intelligent systems that are able to efficiently conform to a particular environment, characterized by a certain kind of available information. Symbolic or well-structured schemes of representation have been studied to encode prior knowledge in a comprehensible form, while the most powerful mechanisms to deal with plain observational data derive from the employment of neurocomputing methods.

In real-world problems, learning flows out from a context where some incomplete (possibly inexact) domain knowledge and a finite amount of data are available. A wide range of possibilities, therefore, exists between pure knowledge-driven and pure data-driven learning. In particular, the process of *knowledge discovery* starting from data is becoming an urging issue whose relevance is growing together with the technological progresses that permit the manipulation of massive amounts of data [14]. Knowledge discovery from data (KDD) basically concerns the study of mechanisms which allow to recognize precise patterns in a set of data that can be exploited as a form of knowledge in particular learning tasks [17]. Actually, KDD strategies reverberate also over a number of data-mining related topics, affecting data pre-processing methods, data clustering, explanation of the obtained querying results.

In this context, the emerging role of fuzzy logic assumes crucial relevance amongst the innovative approaches that try to make use of natural language expressiveness to tackle KDD problems, thus improving comprehensibility of the obtained results. Fuzzy sets lend themselves well to handle incomplete and heterogeneous data and their application to knowledge discovery processes results very helpful in terms of interpretability.

Fuzzy logic methods have been largely combined with different approaches to originate several successfully hybrid systems oriented to exploit in a complementary fashion the individual strong points of various techniques. By following this direction, systems endowed with higher performance and a kind of intelligent behavior can be designed. The knowledge-data trade off has been a major driving force behind *neurosymbolic integration*, a research area currently aimed at building hybrid intelligent systems, commonly known as neuro-fuzzy systems, that realize the synergy of symbolic and neural processing to undertake real-world KDD tasks. From the first pioneering papers about neuro-fuzzy systems [2,25,36,42,45], many other examples of such systems have been developed and successfully used in a wide range of applications [31,44]. An overview about the state-of-art of neuro-fuzzy systems can be found in [1] and [35].

In this paper we illustrate a KDD methodology based on a combination of fuzzy techniques with neural network learning to extract knowledge from data in the form of linguistic rules. The aim of our work is to develop a systematic fuzzy modeling mechanism which is capable of automatically generating a rule-base from numerical data (without any assumption about the structure of the data), finding a suitable rule number, optimizing the parameters of the fuzzy membership functions and providing a readily interpretable model.

The core of the framework is a knowledge extraction procedure that is devoted to identify the structure and the parameters of a fuzzy rule base, through the learning of a neuro-fuzzy network. The learning strategy, that we previously presented in [8] and [9], involves two phases: an unsupervised learning

phase, to group training data into clusters that are used to define fuzzy rules, and a supervised learning, to refine the extracted rules. We integrate the knowledge extraction core with two stages devoted to improve reliability and readability of the extracted knowledge. Namely, a pre-processing performing a selection of input variables and a post-processing leading to a granulation of output values.

The framework is designed to deal with complex multi-input multi-output problems in such a way that two modeling strategies can be employed: it can be derived a single MIMO model, catching altogether the existing relationships among input and output variables, or a collection of MISO models, that separately capture relations between input variables and a single output variable.

To show how complicated problems can be handled by the proposed framework, a complex industrial problem is considered to perform a knowledge discovery process, managing actual experimental data. The problem at hand consists in predicting the chemical composition of ashes deriving from combustion processes for electric generation.

The paper is organized as follows. Section 2 outlines the general scheme of the KDD framework, giving details about the adopted fuzzy inference model and the related neuro-fuzzy architecture. Section 3 describes the knowledge extraction procedure which involves identification and optimization of the structure and the parameters of fuzzy rules. In Section 4 we introduce the pre- and post-processing stages integrated with the core of the proposed methodology. Section 5 is concerned with the application of the proposed framework to the ash property prediction problem. Finally, some conclusive remarks are drawn in Section 6.

2. General scheme of the KDD methodology

In this section we introduce the proposed modeling framework oriented to find patterns and structures in data and to represent them in the form of linguistic rules. In order to obtain readable knowledge from data, the framework involves an integration between a knowledge extraction procedure and two additional processing phases, as depicted in Fig. 1.

The knowledge extraction process, which represents the core of the proposed framework, is performed through the learning of a neuro-fuzzy network that encodes in its structure the discovered knowledge in the form of fuzzy rules. Two further stages are integrated with the knowledge extraction procedure, namely:

- a pre-processing variable selection stage, which aims at simplifying the predictive task by identifying the most relevant input variables for the derivation of each output value;

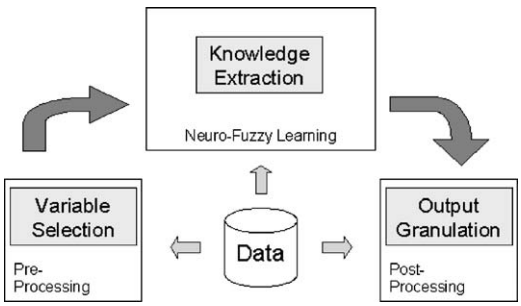

Fig. 1. General scheme of the KDD methodology.

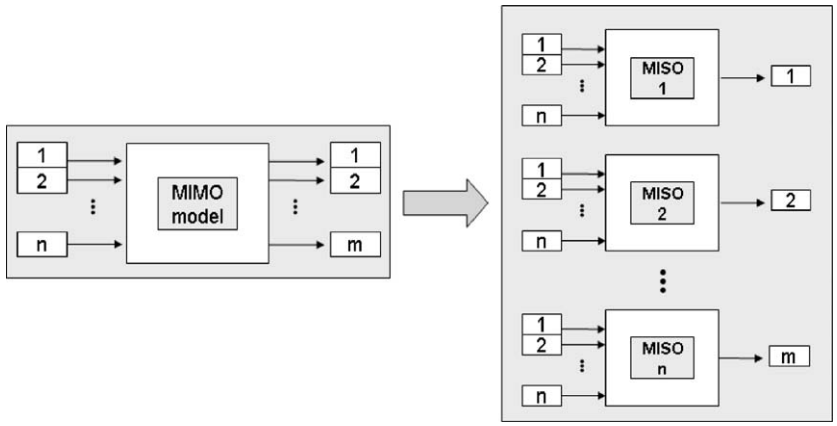

Fig. 2. Splitting of a MIMO problem into a collection of MISO problems.

- a post-processing output granulation stage, which is intended to better specify the final obtained results through the use of prediction intervals.

These processing steps are oriented to further improve the accuracy and comprehensibility of the obtained fuzzy rule base. However it should be emphasized that not all the aforementioned stages are always necessary: some of them can be omitted in specific cases. For example, the input selection could be dropped if the process to be modeled is low-dimensional or if prior knowledge about the relevant input variables is available. The employment of prediction intervals, moreover, finds justification when a measure of the uncertainty on the estimation is needed. Similarly, the employment of prediction intervals can be bypassed when only scalar outputs are required.

The general nature of the proposed framework lends itself to adapt to the particular task at hand. To better deal with complex multi-output problems, the framework offers a twofold choice between the identification of a single MIMO model or a collection of MISO models, as depicted in Fig. 2. In the multi-MISO approach, each model is separately designed and trained according to the proposed KDD methodology.

The choice between the MIMO or multi-MISO approach should be driven by the complexity of the problem to be addressed. In particular, the MIMO approach would provide for a more compact model, especially suitable when the output variables are strongly correlated [40]. On the other hand, by splitting a global multi-output modeling problem into several simpler single-output tasks, we achieve an enlargement of the whole model variance amount due to the increased number of free parameters [20]. Consequently, it is possible to directly intervene in the bias-variance tradeoff in order to build a more flexible model that can be useful in solving highly non-linear prediction problems. Approaches that model complex MIMO systems through identification of a collection of MISO fuzzy models can be found, for example, in [3,19,21,23,24,34], while methods to deal with genuine MIMO models have been developed in [28,47].

Before describing in detail the stages involved in our framework, we outline the basics of the adopted fuzzy reasoning scheme and the related neuro-fuzzy architecture.

Suppose that a set $T = \{(\mathbf{x}_t, \mathbf{y}_t)\}_{t=1}^{N}$ of N input–output data describing the behavior of a process is available. The aim of knowledge extraction is to derive a model that predicts the values of the output variables $\mathbf{y} = (y_1, \ldots, y_m)$, given the values of the input variables $\mathbf{x} = (x_1, \ldots, x_n)$. In other words, we

have to approximate the unknown function:

$$f : X \subset \Re^n \to Y \subset \Re^m. \tag{1}$$

If a multi-MISO strategy is adopted, the dependency (1) can be modeled as a collection of m separate dependencies:

$$f_j : X \subset \Re^n \to Y \subset \Re \quad j = 1, \ldots, m. \tag{2}$$

For the sake of simplicity, henceforth we describe the adopted fuzzy model as a MISO model, since an extension to an MIMO model is straightforward. Without loss of generality, we refer to the modeling task of a dependency f_j by means of a fuzzy model M_j dropping the index j. In particular, we consider a fuzzy model based on rules of the following type:

$$\text{IF } (x_1 \text{ is } A_{1r}) \text{ AND } \cdots \text{ AND } (x_n \text{ is } A_{nr}) \text{ THEN } (y \text{ is } b_r)$$

where $r = 1, \ldots, R$ (being R the number of fuzzy rules of the model); A_{ir} $(i = 1, \ldots, n)$ are fuzzy sets defined over the input variables x_i; b_r is a fuzzy singleton defined over the output variables y. Each fuzzy set A_{ir} is represented by a Gaussian membership function in the form

$$\mu_{ir}(x_i) = \exp\left(-\frac{(x_i - c_{ir})^2}{2a_{ir}^2}\right), \tag{3}$$

where c_{ir} and a_{ir} are the center and the width of the Gaussian function, respectively.

Based on a set of R rules (generated through the proposed modeling framework), the output of the fuzzy model for any unknown input vector $\mathbf{x}_0$ is obtained adopting the following fuzzy reasoning procedure:

(1) Calculate the matching degree of input $\mathbf{x}_0$ to the rth rule, for $r = 1, \ldots, R$, by means of Larsen product operator:

$$\mu_r(\mathbf{x}_0) = \prod_{i=1}^{n} \mu_{ir}(x_{0i}) \quad r = 1, \ldots, R. \tag{4}$$

(2) Calculate the inferred output $\hat{y}_0$ as:

$$\hat{y}_0 = \frac{\sum_{r=1}^{R} b_r \mu_r(\mathbf{x}_0)}{\sum_{r=1}^{R} \mu_r(\mathbf{x}_0)}. \tag{5}$$

Summarizing, the knowledge extraction process can be stated as that of finding a proper number R of fuzzy rules and optimal parameters $(\mathbf{c}, \mathbf{a}, \mathbf{b})$ from a data set T of input–output pairs, such that the fuzzy model closely approximates the unknown function $f(\cdot)$. To accomplish this job, the fuzzy model is implemented as a particular neural network, so that structure and parameter identification of the fuzzy rule base can be obtained by adapting the topology and the parameters of the corresponding neuro-fuzzy network, in a data-driven fashion.

The considered neuro-fuzzy network has a three-layer architecture. According to the layer they belong to, units in the network have the following specifications:

- Units in the first layer L_1 receive the input values $(x_1, \ldots, x_n)$ and act as membership functions representing fuzzy sets of the corresponding input variable. Units in this layer are arranged into R groups, one for each fuzzy rule. The rth group includes n units, that correspond to the input fuzzy sets

defining the premise part of the rth rule. Each unit $u_{ir} \in L_1$ receives the input value x_i and computes the membership value $\mu_{ir}(x_i)$. According to (3), the output of unit $u_{ik} \in L_1$ is given by the following function:

$$O_{ir}^{(1)} = \exp\left(-\frac{(x_i - c_{ir})^2}{2a_{ir}^2}\right) \quad i = 1, \ldots, n, \quad r = 1, \ldots, R, \tag{6}$$

where c_{ir} and a_{ir} constitute the free parameters of unit $u_{ir} \in L_1$.

- The second layer L_2 contains R units that perform precondition matching of fuzzy rules. These units are fixed, meaning that no modifiable parameter is associated with them. For each unit, there are n fixed links deriving from the preceding layer L_1, representing the IF-part of the fuzzy rule. The output of unit $u_r \in L_2$ is computed according to the expression of the rule activation strength in (4), namely:

$$O_r^{(2)} = \prod_{i=1}^{n} O_{ir}^{(1)} \quad r = 1, \ldots, R. \tag{7}$$

- The third layer L_3 provides the output value $\hat{y}$ resulting from the inference of rules, according to (5):

$$\hat{y} = O^{(3)} = \frac{\sum_{r=1}^{R} b_r O_r^{(2)}}{\sum_{r=1}^{R} O_r^{(2)}}. \tag{8}$$

Connections between layer L_2 and L_3 are weighted by the fuzzy singletons b_r that represent another set of free parameters for the neuro-fuzzy network.

The architecture of the neuro-fuzzy network is depicted in Fig. 3, where nodes representing the premise part of a fuzzy rule are enclosed in a gray circle, that can be regarded as a *meta-node* of the network. The weights of the network correspond to the Gaussian membership function parameters c_{ir}, a_{ir} and to the consequent singletons b_r. In other words, each meta-node r is associated with two premise weight vectors $\mathbf{c}_r$ and $\mathbf{a}_r$ and one consequent weight vector $\mathbf{b}_r$.

This neuro-fuzzy network encodes a set of fuzzy rules in its topology, and processes information so as to match the adopted fuzzy reasoning scheme. Hence, it can be regarded both as an adaptive fuzzy

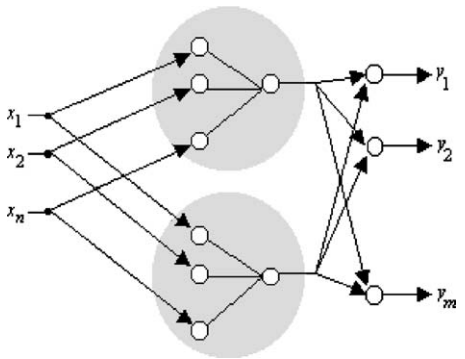

Fig. 3. The neuro-fuzzy network.

inference system with the capability of learning fuzzy rules from data, and as a connectionist architecture provided with linguistic meaning.

3. Knowledge extraction by neuro-fuzzy learning

To extract knowledge from data in the form of fuzzy rules, we apply a hybrid learning procedure to the neuro-fuzzy network, whose preliminary scheme has been previously presented in [8,9]. The knowledge extraction procedure involves two learning steps (Fig. 4). In the first step, an unsupervised learning scheme is applied to identify the structure and the parameters of fuzzy rules. In the second step, fuzzy rule parameters are tuned via supervised learning to improve the accuracy of the derived knowledge. Unlike other neuro-fuzzy approaches based on similar two-phase learning strategies [12,16,27,29,41], the first phase of our approach can find automatically a suitable number of clusters, hence the proper number of rules for the problem at hand. In the following each learning phase is detailed.

3.1. Structure and parameter identification

In the first learning step, the proper number of fuzzy rules and membership functions (size of the network topology), together with initial values of rule parameters (network weights) are simultaneously determined, thereby enabling the construction of a rule base in a self-organized fashion. This is accomplished by clustering the input space and then deriving a fuzzy rule from each cluster. To find proper prototypes of clusters in the input space, a modified competitive learning algorithm, similar to those proposed in [32] and [46], is defined to learn the weight vectors $\mathbf{c}_r$ on the basis of available input data. To start this learning phase, an initial structure of the neuro-fuzzy network is firstly constructed based on a guessed maximum number R^* of rules (meta-nodes) given as a form of a priori knowledge. Then, during learning, the network self-organizes its structure via a mechanism that gradually drives the weight vectors of extra meta-nodes far away from the distribution of the data, thus reducing the final number of clusters. In this way the appropriate number of rules for representing the input data is automatically selected.

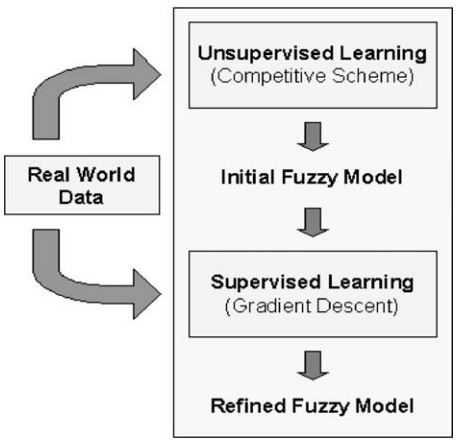

Fig. 4. Scheme of the neuro-fuzzy learning.

This clustering procedure, though implemented in an unsupervised context, is similar to a cluster repulsion mechanism described in [43] which finds application when a priori class information knowledge is available. However, the key point of our algorithm is that no expert intervention is needed for the completion of the competitive learning and the most appropriate number of clusters (corresponding to the number of fuzzy rules) is determined merely from data investigation.

Only meta-nodes of the network are involved in the competitive learning phase. When an input vector $\mathbf{x}$ is presented, such nodes compete and the meta-node whose weight vector $\mathbf{c}$ is closest to the input vector is chosen as winner, while the second closer node is marked as the *rival*. Then, the weight vector of the winner node is rewarded, i.e. updated so as to become closer to the current input vector, while the weight vector of the rival is punished, i.e. updated so as to move it away from $\mathbf{x}$. This mechanism tries to push the weight vector of the rival node far away from the cluster towards which the weight vector of the winner is moving, thus implicitly making sure that each cluster is represented by only one weight vector. As stated also in [46], the use of such reward/punishment mechanism, that gradually drives the weight vectors of useless nodes far away from the distribution of the data, allows the number of necessary meta-nodes, and hence the number of rules, to be automatically selected. The complete competitive learning algorithm is summarized below.

Set $\tau := 0$;
Initialize randomly the center vectors $\mathbf{c}_r^{(\tau)} (r = 1, \ldots, R^*)$.
Initialize the learning rates α_ω and α_ρ for the winner and the rival, respectively, so that $0 \leqslant \alpha_\rho \leqslant \alpha_\omega \leqslant 1$.
Repeat
(1) For each input vector $\mathbf{x}_t$, $t = 1, \ldots, N$ perform the following steps:
(a) Compute the distances:

$$d(\mathbf{x}_t, \mathbf{c}_r^{(\tau)}) = \frac{n_r}{\sum_{s=1}^{R^*} n_s} d_E(\mathbf{x}_t, \mathbf{c}_r^{(\tau)}) \quad r = 1, \ldots, R^*,$$

where n_r is the cumulative number of the winning occurrences for meta-node r and $d_E(\cdot, \cdot)$ is the Euclidean distance.
(b) Determine the winning meta-node ω and its rival ρ using the rule:

$$\omega = \arg \min_r d(\mathbf{x}_t, \mathbf{c}_r^{(\tau)}) \quad \rho = \arg \min_{r \neq \omega} d(\mathbf{x}_t, \mathbf{c}_r^{(\tau)}).$$

(c) Update the number of winning occurrences for the winner:

$$n_\omega := n_\omega + 1.$$

(d) Update weight vectors of the winning and the rival meta-node according to:

$$\mathbf{c}_\omega^{(\tau+1)} := \mathbf{c}_\omega^{(\tau)} + \alpha_\omega \left(\mathbf{x}_t - \mathbf{c}_\omega^{(\tau)} \right) \quad \mathbf{c}_\rho^{(\tau+1)} := \mathbf{c}_\rho^{(\tau)} - \alpha_\rho \left(\mathbf{x}_t - \mathbf{c}_\rho^{(\tau)} \right)$$

(2) Set $\tau := \tau + 1$;
Until $\frac{1}{R^*} \sum_{r=1}^{R^*} \| \mathbf{c}_r^{(\tau)} - \mathbf{c}_r^{(\tau-1)} \| \leqslant \varepsilon$
Remove all meta-nodes whose weight vector falls outside the input range.

Such competitive learning performs clustering in the input space with the ability of adapting the number of clusters as the learning proceeds. Starting with R^* meta-nodes, the network self-organizes its structure by automatically finding a set of R meta-nodes ($R \leqslant R^*$) whose weight vectors $\mathbf{c}_r$ ($r = 1, \ldots, R$) represent

the centers of spherical clusters in the input space. Each cluster is regarded as a multi-dimensional fuzzy set representing the antecedent of a fuzzy rule. Precisely, the cluster prototype vector $\mathbf{c}_r = (c_{1r}, c_{2r}, \ldots, c_{nr})$ provides centers of the Gaussian membership functions μ_{ir}, while the widths $\{a_{ir}\}$ are defined using the *first-nearest-neighbor* heuristic: $a_{ir} = \frac{\|\mathbf{c}_r - \mathbf{c}_s\|}{\gamma}$, where $\mathbf{c}_s$ is the cluster center nearest to $\mathbf{c}_r$ and γ is an *overlap* parameter ranging in [1.0, 2.0]. The consequents parameters b_r are obtained as follows:

$$b_r = \frac{\sum_{t=1}^{N} \mu_r(\mathbf{x}_t) y_t}{\sum_{t=1}^{N} \mu_r(\mathbf{x}_t)} \tag{9}$$

with $\mu_r(\mathbf{x}_t)$ being the matching level of the premise part of the rule defined as in (4), which can be computed once premise membership functions μ_{ir} have been derived as described above.

3.2. Parameter optimization

After structure identification, we obtain both the number and the initial parameters of rules that are used to build the fuzzy models. To improve accuracy, i.e. to find the best fit to the data, in the second learning phase all the parameters of the fuzzy rules are finely tuned via a gradient descent technique. A very large number of neuro-fuzzy systems are based on back-propagation-like algorithms to optimize the parameters of fuzzy systems, from the most famous ANFIS [25] to its variants proposed in [22,29,38,39,45]. Similarly, in this work, a supervised learning algorithm based on a gradient-descent technique is formulated, that optimally adjusts the parameters c_{ir}, a_{ir}, b_r using the mean squared error (MSE) as performance index. The update formulas for the parameter learning algorithm are derived as:

$$\Delta b_r = -\eta \frac{\partial E}{\partial b_r} = \eta \delta^{(3)} O_r^{(2)}, \tag{10}$$

$$\Delta c_{ir} = -\eta \frac{\partial E}{\partial c_{ir}} = -\eta \delta_r^{(2)} \frac{x_i - c_{ir}}{a_{ir}^2}, \tag{11}$$

$$\Delta a_{ir} = -\eta \frac{\partial E}{\partial a_{ir}} = -\eta \delta_r^{(2)} \frac{(x_i - c_{ir})^2}{a_{ir}^3}, \tag{12}$$

where $\delta^{(3)} = -\frac{y - O^{(3)}}{\sum_{s=1}^{R} O_s^{(2)}}$ and $\delta_r^{(2)} = \delta^{(3)}(b_r - O^{(3)}) O_r^{(2)}$; η is the learning rate.

Once the learning is completed, the network architecture encodes the knowledge learnt in the form of fuzzy rules and processes data following fuzzy reasoning principles.

4. Readability and reliability improvement

The neuro-fuzzy learning procedure described above is a valid tool to extract accurate knowledge from data. Nevertheless, it does not take into account the issue of readability of the fuzzy rules. Actually, the resulting rules may reveal too complex to be read, thus missing the benefits in terms of comprehensibility deriving from the employment of a fuzzy representation of knowledge. In order to increase the interpretability of fuzzy rules while preserving (and possibly improving) the accuracy, the proposed KDD methodology offers two further stages besides the neuro-fuzzy learning core.

On one side, a pre-processing stage consisting in an input variable selection procedure is integrated in the KDD methodology. Indeed, one main factor that influences the readability of the extracted knowledge is the number of input variables: each rule should use as few variables as possible to be more comprehensible. Moreover, it is plausible that taking into account the entire set of input variables to predict a particular output variable could not prove to add information, but often may increase noise.

Moreover, when a high degree of variability in output values is experienced, a simple scalar output may not sufficiently provide useful information. A post-processing phase, based on the definition and the employment of prediction intervals, contributes to an output granulation that allows a better understanding and an uncertainty estimation of the inferred information.

4.1. Variable selection

Variable or feature selection is a pre-processing step of prominent importance in KDD. From the theoretical side, feature selection is necessary to alleviate the so-called "curse of dimensionality", which implies an exponential growth of examples required to represent a function as the number of features increases [4]. Curse of dimensionality is particularly felt in data mining contexts, where data are often defined by a high number of features, thus requiring a huge number of records that are usually not available. From a more practical standpoint, feature selection can drastically reduce the computation burden required for the optimization of the fuzzy model, due to a relevant reduction of free parameters.

The problem of feature selection is intrinsically difficult, since an exhaustive search of a proper subset of features has a combinatorial nature. As a consequence, algorithms for searching a proper subset of features are often time consuming and usually return only suboptimal solutions (see e.g. [13,18,26]).

In this work we employ an alternative approach that consists in ranking features according to an importance factor. The adopted procedure, similar to the one proposed in [30], operates by defining a number of fuzzy models with an increasing number of input variables that are selected from a list of features sorted according to their rank. Finally, the fuzzy model of highest quality is chosen, and the corresponding features are hence selected.

In order to assign a rank value to each feature, an initial fuzzy model that incorporates all possible input variables is built from data by a fast clustering algorithm illustrated below.

Given a data set $T = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N | \mathbf{z}_t = (\mathbf{x}_t, y_t)\}$

(1) Set the first prototype $\mathbf{w}_1 := \mathbf{z}_1$
(2) Set $\tau := 1, \quad N_{S_1} := 1$
(3) For each $t = 1, 2, \ldots, N$,
  (a) Find the nearest prototype $\mathbf{w}_k$ such that
    $\|\mathbf{w}_k - \mathbf{z}_t\| = \min_{s=1,\ldots,N} \|\mathbf{w}_s - \mathbf{z}_t\|$
  (b) If $\|\mathbf{w}_k - \mathbf{z}_t\| \leqslant \delta$ (where δ is a predefined threshold), then
    (i) Set $N_{S_k} := N_{S_k} + 1$
    (ii) Set $\mathbf{w}_k := \mathbf{w}_k + \|\mathbf{w}_k - \mathbf{z}_t\|/N_{S_k}$
  (c) else
    (i) Set $\tau := \tau + 1$
    (ii) Create a new prototype $\mathbf{w}_\tau = \mathbf{z}_t$
(4) End for

Such algorithm returns a set of prototypes in the input/output product space, by performing only one cycle on the data set. Each discovered prototype $\mathbf{w}_r = (w_{1r}, \ldots, w_{nr}, w_{n+1,r})$ corresponds to one fuzzy rule as follows:

$$\text{IF } x_1 \text{ is } A(w_{1r}, \sigma) \text{ AND } \ldots \text{ AND } x_n \text{ is } A(w_{nr}, \sigma) \text{ THEN } y = w_{n+1,r},$$

where the fuzzy sets $A(w_{ir}, \sigma)$ are characterized by Gaussian membership functions of center w_{ir} and width σ. The common width σ is selected by choosing the value that provides better accuracy results among a set of possible widths.

The derived fuzzy model is used to rank the input variables by assessing the sensibility of the model output with respect to changes of each input variable. Specifically, to assess the ranking of the ith feature, we consider a simplified fuzzy model with rules of the form:

$$\text{IF } x_i \text{ is } A(w_{ir}, \sigma) \text{ THEN } y = w_{n+1,r}.$$

The available training set T is applied on the simplified rule base in order to evaluate the contribution of the ith input variable to the output variable. Given the N output values $\hat{y}(x_{ti}),\ t = 1, \ldots, N$ provided by the simplified fuzzy model, the importance factor of the variable x_i is measured by the following index, as suggested in [30]:

$$I_i = \max_{t=1,\ldots,N} \left|\hat{y}(x_{ti})\right| - \min_{t=1,\ldots,N} \left|\hat{y}(x_{ti})\right| \tag{13}$$

corresponding to the contribution of the ith component of the tth input vector in T.

The importance factor provides a measure of variability of the output variable by varying the ith input variable. More specifically, the higher is the importance factor, the wider is the range covered by the output variable when the ith input variable fluctuates in its domain. As a consequence, the higher is the influence of the ith input variable in determining the final output value. As a limit case, if the importance factor is zero then the output value is not affected by any specific value of the input variable, i.e. the input variable does not contribute to determine the final output value.

All the input variables are evaluated by the same procedure and then sorted according to their importance factor I_i. The list of sorted variables can be examined by domain experts to discard useless variables, or conversely to keep those that are judged necessary for the model. Successively, a correlation analysis can be useful to remove from the list all variables that are highly correlated with other variables of higher rank. Such filtered list is used as a starting point for an automatic feature selection procedure aimed at selecting the subset of variables that provides a good accuracy of the final fuzzy model.

To avoid combinatorial explosion, the ranking order of the feature list is exploited. Specifically, given the feature list $(x_{i_1}, x_{i_2}, \ldots, x_{i_v}),\ v \leqslant n$, a sequence of v fuzzy models is built according to the following rule schema:

$$\text{IF } x_{i_1} \text{ is } A(w_{i_1,r}, \sigma) \text{ AND } x_{i_2} \text{ is } A(w_{i_2,r}, \sigma) \ldots \text{ AND } x_{i_h} \text{ is } A(w_{i_h,r}, \sigma)$$
$$\text{THEN } y_1 = w_{n+1,r},$$

where $h \leqslant v$. As a consequence, the hth fuzzy model is built from the first h variables contained in the feature list. If some features are forced to be present in the fuzzy model, they can be placed at the top of the list. Each fuzzy model is evaluated on a test set of data and the model with highest accuracy is selected (for equal accuracies, the model with fewer variables is selected). As a result, the corresponding subset of features is retained and employed for the design of the definitive fuzzy model according to the proposed framework.

4.2. Output granulation

To enable a better understanding of the extracted fuzzy rules, the knowledge extraction procedure is equipped with a granular representation of the estimated outputs, which embodies a measure of uncertainty on the estimation for a user-defined confidence level. This granulation is achieved through the definition of prediction intervals that provide a range of variability for the inferred output values.

Given a fuzzy rule base derived through the knowledge extraction procedure described in Section 3, the calculation of prediction intervals starts from evaluating the errors on the training data:

$$e_t = \hat{y}_t - y_t \quad t = 1, 2, \ldots, N. \tag{14}$$

The values e_t, $t = 1, 2, \ldots, N$ can be considered as an independent and identically distributed sampling of a random variable e of unknown distribution. As a consequence, the mean value:

$$\bar{e} = \frac{1}{N} \sum_{t=1}^{N} e_t \tag{15}$$

approaches the normal distribution for large N. For such a normally distributed random variable, prediction intervals can be calculated.

A prediction interval $[L_\alpha, U_\alpha]$ of confidence level α is an interval that will include the error e_{new} for a newly drawn example $\mathbf{x}_{\text{new}}$ with probability greater than $1 - \alpha$ (see [37]). Formally:

$$\wp(e_{\text{new}} \in [L_\alpha, U_\alpha]) \geqslant 1 - \alpha \tag{16}$$

or, equivalently:

$$\wp(y_{\text{new}} \in [\hat{y}_{\text{new}} - U_\alpha, \hat{y}_{\text{new}} - L_\alpha]) \geqslant 1 - \alpha. \tag{17}$$

The relation (17) defines a statistical method to estimate the true value of the underlying function approximated by the fuzzy rule base with a desirable confidence level. The extreme values L_α and U_α of the prediction interval are defined as follows:

$$L_\alpha = \bar{e} - t_{\frac{\alpha}{2},N-1} \left(s\sqrt{\frac{1}{N} + 1} \right), \tag{18}$$

$$U_\alpha = \bar{e} + t_{\frac{\alpha}{2},N-1} \left(s\sqrt{\frac{1}{N} + 1} \right), \tag{19}$$

where $t_{\frac{\alpha}{2},N-1}$ is the value of the Student distribution with $N - 1$ degrees of freedom corresponding to the critical value $\frac{\alpha}{2}$, and s is the sampled standard deviation

$$s = \sqrt{\frac{1}{N-1} \sum_{t=1}^{N} (e_t - \bar{e})^2}. \tag{20}$$

The width of the prediction interval provides a measure of the accuracy of the knowledge base. The less accurate the knowledge base, the wider the prediction interval is. Hence, prediction intervals are introduced in our framework to increase reliability of the extracted model. Precisely, a prediction interval defined through (18) and (19) on the basis of the inferred outputs $\hat{y}_t$ is calculated, thus providing a range of variability for the outcome of the model.

In addition, in order to improve readability of the extracted knowledge base, prediction intervals can also be calculated for each rule consequent, leading to fuzzy rules of the following form:

$$\text{IF } x_1 \text{ is } A_{1r} \text{ AND } x_2 \text{ is } A_{2r} \text{ AND } \ldots \text{ AND } x_n \text{ is } A_{nr}$$
$$\text{THEN } y \in [b_r - U_\alpha, b_r - L_\alpha]\ (1-\alpha).$$

The insertion of prediction intervals into fuzzy rules provides a more explanatory knowledge base, since they can help users to perceive an entire range of validity rather than single numerical values. However, it should be noted that such intervals are used only for knowledge representation purpose, and they are not employed in the inference process, which is carried out by means of singletons outputs, according to [30].

5. A real application example

The core of the proposed framework, namely the knowledge extraction process, has been employed in a wide variety of problems. Among these, we could mention the rule extraction from data in the field of image processing [6,10], classification [8,9] and system identification [7]. To highlight some of the most interesting features of the whole KDD framework, here we show the application to a real problem concerning the prediction of ash properties deriving from combustion processes for electric generation. The predictive problem can be explicitly stated as follows: given a certain kind of combustible substances which undergo a specific combustion process, we want to determine the chemical properties of the resulting ashes.

In the field of electric generation an issue of increasing relevance is represented by the analysis of the discarded materials deriving from the combustion processes carried out inside the power plants. The awareness about the after-effects of the chemical emissions affecting the environment and the human health has notably progressed in the last decades, urging a considerable research effort toward the industrial dross analysis. Furthermore, considerations about the reusability of particular classes of discarded materials provide also an economic incentive for this kind of investigation. In particular, the prediction of chemical composition of ashes deriving from combustion processes is becoming a strategic and challenging problem, both for environmental impact evaluations and for the possibility of recycling ashes in the cement production. For many years the industrial research community in the field of electric generation has been developing methods for predicting the ash properties and much of this work has concentrated on “traditional” techniques (e.g. numerical treatment of differential equations in dynamical systems). The application of neuro-fuzzy approaches to tackle such problem provides the additional advantage to model the relationships underlying the input–output data by means of fuzzy rules that express the extracted knowledge in a transparent and interpretable way (see for example [11]).

Although this prediction problem belongs to the classical field of system identification, it is well suitable to evaluate the effectiveness of our framework in its resolution. Indeed, it is especially complex due to

the multi-variable physical process to be modeled, the heterogeneous nature of the involved variables and the incompleteness of the available dataset.

5.1. Dataset description

The dataset, provided by *ENEL Produzione e Ricerca S.p.A.* (*Italy*), collects information deriving from a number of observations acquired during actual combustion processes carried out in a number of plants located in Italy.

Besides the intrinsic difficulties of the predictive problem at hand, concerning a complex physical process with several variables, an additional troublesome issue derived from the small size of the dataset comprising only 54 samples. Each of them pertains to the measurements related to a combustion process performed at some detailed conditions, making use of some kind of fuel (or combination of mixed fuels), with particular plant arrangements, referring to distinct classes of resulting ashes.

Every sample in the dataset collects 32 input features and 22 output values. Precisely, the input variables encode the following characteristics:

- kind of combustion (single fuel or mixed fuels);
- electric load of the power plant;
- drawing point of ashes;
- chemical composition of fuels.

All the input variables take on numerical values, except for the kind of combustion and the drawing source variables, which are characterized by discrete categorical values. Such categorical variables have been handled like the other variables by assigning to each category a distinct value in $[0, 1]$. The 22 output variables represent the values to be predicted and they refer to 22 distinct chemical compounds affecting the type of ashes we are considering.

The analysis of the available dataset pointed out two issues. The first concerns the presence of incomplete samples. Incomplete data represent a common occurrence in real-world applications: to tackle this kind of problem different approaches have been proposed in literature [5]. If the quantity of data is sufficiently large and the proportion of affected patterns is small, then the simplest solution is to discard incomplete samples from the dataset. This particular strategy, besides the inherent drawback of modifying the effective data distribution (mostly in cases where the missing values are not uniformly distributed in the dataset), is totally inappropriate in our case, since we are dealing with a small amount of samples. For this reason, we applied a "fill in" approach consisting in replacing missing values with mean values estimated on the basis of complete samples.

The second issue regards the heterogenous nature of data. The remarkable differences between the ranges of some variables, often reported into the dataset with disparate measuring units, reflect the extreme heterogeneity of data representing very different types of features. This kind of problem could largely affect the knowledge extraction process; this is the reason why it is necessary to scale the variable values. A standard scaling procedure has been employed: data were centered and standard deviation was made equal to 1. Defining by $\bar{z}_i$ and σ_i, respectively, the mean and the variance evaluated for the ith variable over N samples, each variable z_i^t of the dataset has been transformed into $\tilde{z}_i^t$ by:

$$\tilde{z}_i^t = \frac{z_i^t - \bar{z}_i}{\sigma_i} \quad t = 1, \ldots, N, \quad i = 1, \ldots, M,$$

where M is the total number of variables (input and output).

The dataset obtained after these pre-processing steps was employed both to perform the knowledge extraction process by neuro-fuzzy learning and to improve the readability of the extracted knowledge, as described in the following section.

5.2. Experimental results

To perform simulations, a training set composed of 31 elements was derived from the pre-processed dataset and used to extract fuzzy rule bases via the proposed neuro-fuzzy learning. The remaining 23 samples constituted the test set to be exploited in checking the generalization ability of the obtained predictive models. To better assess the performance of the extracted models, five different partitions of the dataset in a training and test set were considered.

Firstly, the core of the methodology (plain knowledge extraction) was employed to produce both a MIMO model and a set of 22 MISO models. In the MIMO approach, five single models embedding the relationships between the input variables and all the output variables were derived. The extraction of the fuzzy rule bases was carried out by training a proper neuro-fuzzy network using the two-phase learning scheme described in Section 3.1. The application of the competitive learning phase produced initial rule bases comprising a number of rules ranging from 8 to 14, starting from a number of 20 rules (clusters) given as a form of a priori knowledge. Then, the rule bases were refined through 1000 epochs of supervised learning. Table 1 reports the average predictive accuracy in terms of MSE over training and test set for each output of the MIMO models.

Also, the plain knowledge extraction procedure was used as a multi-MISO strategy to build five multi-MISO models, i.e. five collections of 22 multi-input single-output fuzzy rule bases. Each rule base, devoted to the prediction of a single output variable, was derived using the same learning setup adopted in the generation of the MIMO models. Consequently, the MISO models are characterized by a number of fuzzy rules ranging from 8 to 14, too. Table 2 reports the average predictive accuracy in terms of the standard MSE over training and test set for each MISO model.

As it can be seen from Tables 1 and 2, all the extracted fuzzy rule bases exhibit a satisfactory prediction accuracy, despite the complexity of the problem and the very limited number of available samples.

Table 1
Average accuracy of MIMO models derived by plain knowledge extraction

Output variable	MSE		Output variable	MSE	
	Training set	Test set		Training set	Test set
1 Al	0.22	0.23	12 Cr	0.24	0.26
2 Ca	0.32	0.34	13 Cu	0.23	0.26
3 Fe	0.28	0.31	14 K	0.19	0.20
4 Mg	0.24	0.26	15 Mn	0.37	0.39
5 Na	0.16	0.17	16 Ni	0.28	0.31
6 P	0.39	0.41	17 Pb	0.29	0.32
7 S	0.50	0.53	18 Sr	0.28	0.31
8 Si	0.27	0.29	19 V	0.24	0.26
9 Ti	0.22	0.23	20 Zn	0.20	0.21
10 Ba	0.25	0.27	21 D50	0.27	0.30
11 Co	0.16	0.17	22 LOI	0.31	0.33

Table 2
Average accuracy of multi-MISO models derived by plain knowledge extraction

MISO models	MSE		MISO model	MSE	
	Training set	Test set		Training set	Test set
1 Al	0.20	0.20	12 Cr	0.27	0.28
2 Ca	0.30	0.34	13 Cu	0.14	0.23
3 Fe	0.11	0.32	14 K	0.24	0.18
4 Mg	0.21	0.22	15 Mn	0.27	0.22
5 Na	0.11	0.11	16 Ni	0.11	0.29
6 P	0.32	0.32	17 Pb	0.40	0.34
7 S	0.35	0.45	18 Sr	0.24	0.24
8 Si	0.22	0.23	19 V	0.15	0.10
9 Ti	0.14	0.12	20 Zn	0.27	0.22
10 Ba	0.26	0.28	21 D50	0.20	0.12
11 Co	0.23	0.15	22 LOI	0.25	0.26

Table 3
Comparison between different predictive models

Approach	Training set	Test set
MIMO models	5.91	6.36
Multi-MISO models	4.99	5.22
Empirical models	6.14	7.09

However, to enable a fair comparison between the obtained results, the predictive errors resulting from MIMO and multi-MISO approaches have been evaluated in terms of Global Mean Squared Error (GMSE), i.e. the sum of 22 MSEs computed for each output variable:

$$\mathrm{GMSE} = \sum_{j=1}^{22} \mathrm{MSE}_j = \frac{1}{N} \sum_{j=1}^{22} \sum_{t=1}^{N} (y_j(t) - \hat{y}_j(t))^2.$$

The comparative results are reported in Table 3. It is evident that, in this particular problem, the multi-MISO strategy leads to an improvement in terms of accuracy with respect to the single MIMO model. Moreover the table reports also the predictive results obtained by a specific model usually employed by ENEL to estimate the output values in this predictive problem. Such model, referred to as Empirical model, estimates the most likely value for each output variable taking essentially into account the chemical composition of the fuel used in the combustion process. Not only such model performs worse than the MIMO and multi-MISO models in terms of GMSE, but it does not exhibit the powerful knowledge representation capabilities which are characteristic of fuzzy rule-based models. For fair comparison, in all the above-described simulations the pre- and post-processing stages of the KDD methodology have not been executed.

To verify the effectiveness of whole methodology in obtaining from data such a knowledge that could prove to be not only accurate, but also understandable, the variable selection and output granulation steps have been integrated with the knowledge extraction procedure. In this second simulation, we followed only the multi-MISO strategy, which seems to be more appropriate for the problem at hand.

Table 4
Accuracy of MISO models derived by integration of variable selection and knowledge extraction

MISO model	#	#	MSE		MISO model	#	#	MSE	
	Input var.	Rules	Training set	Test set		Input var.	Rules	Training set	Test set
1 Al	4	10	0.05	0.06	12 Cr	11	14	0.10	0.26
2 Ca	9	11	0.29	0.14	13 Cu	10	10	0.13	0.16
3 Fe	8	16	0.05	0.17	14 K	2	10	0.06	0.47
4 Mg	6	13	0.10	0.11	15 Mn	6	9	0.33	0.21
5 Na	3	11	0.03	0.21	16 Ni	10	8	0.06	0.26
6 P	11	12	0.16	0.19	17 Pb	9	12	0.26	0.26
7 S	4	11	0.17	0.32	18 Sr	8	11	0.25	0.23
8 Si	3	16	0.1	0.23	19 V	3	7	0.03	0.08
9 Ti	3	12	0.05	0.07	20 Zn	10	11	0.06	0.06
10 Ba	10	12	0.13	0.20	21 D50	8	8	0.18	0.17
11 Co	8	10	0.02	0.03	22 LOI	10	9	0.16	0.27

Firstly, before extracting the multi-MISO models by neuro-fuzzy learning, the variable selection stage was applied to define a set of significant input variables for each output variable, as described in Section 4.1. It should be noted that no information about the relationships between the input and output data was available. Therefore no assumption could be made about the relevance of each input variable for the prediction of a specific output variable. Then, the ensemble of MISO fuzzy models was derived by the knowledge extraction process: while the application of plain knowledge extraction keeps unchanged the number of input variables (32 values) and the structure of each rule base, the integration with the pre-processing phase produced rule bases differing in the number of input variables and in the number of rules, providing simpler MISO models. Table 4 reports the accuracy values of the derived models in terms of MSE, together with the resulting numbers of input variables and rules. As an additional feature, the integration with variable selection improved accuracy of most final models, compared with the results reported in Table 2.

For illustrative purpose, we considered the multi-MISO model that provided the best trade-off between accuracy and model complexity (number of rules and inputs). Fig. 5 depicts the testing error of some MISO models of such collection, specifically the models that predict output variables Aluminum, Calcium, Magnesium, Titanium, Barium, Cobalt, Vanadium and Zinc. In the plots dashed lines indicate the test errors of the model derived by plain knowledge extraction, while solid lines provide the errors of the model resulting from the integration of the knowledge extraction core with variable selection. Also, the extremes of the derived prediction intervals are plotted in dotted lines.

It can be seen that the models derived with the application of variable selection exhibit a better accuracy in comparison to models derived without variable selection. Also, after variable selection, each derived model provided output values that fall mostly within the prediction interval computed for the related output variable.

Finally, to show how the application of the pre- and post-processing stages leads to an improvement in readability of the extracted knowledge, in Fig. 6 we report the fuzzy rule base related to a specific output variable (Vanadium). Only three input variables (Cu, V and DRAWING point) are involved in the prediction of the Vanadium. Each input variable in rule antecedents is expressed by 0.5-cuts of fuzzy sets. In the case of DRAWING variable, the 0.5-cut produces a set of discrete values since this particular

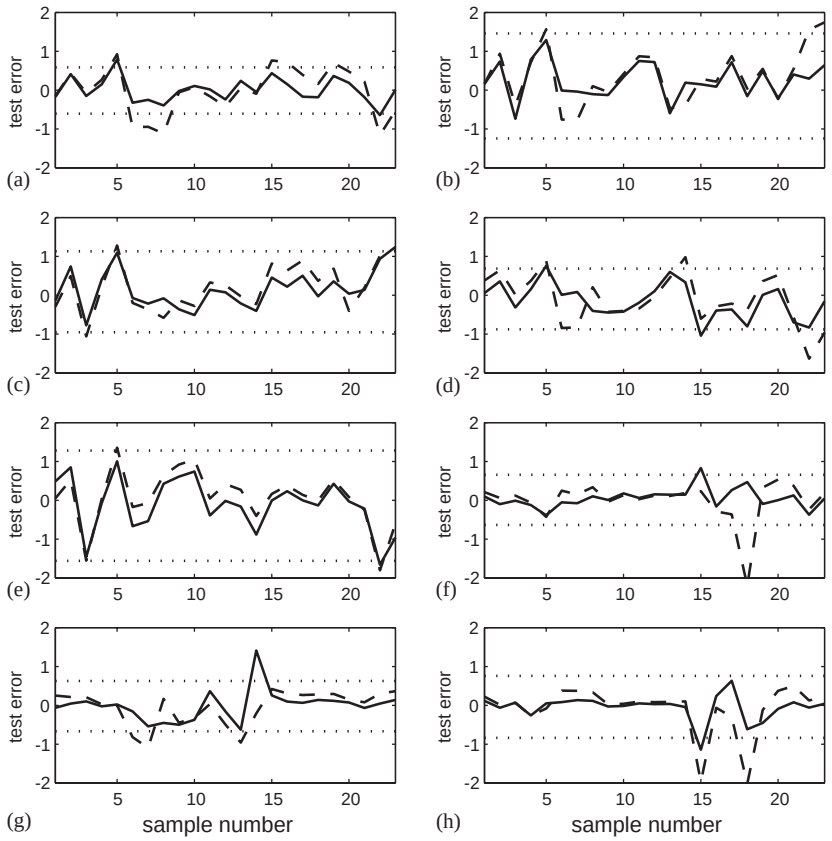

Fig. 5. Test prediction errors obtained by the application of plain knowledge extraction (dashed line) and knowledge extraction with variable selection (solid line) concerning the output variables Al (a), Ca (b), Mg (c), Ti (d), Ba (e), Co (f), V (g), Zi (h). The horizontal lines represent the prediction intervals for each output variable.

variable is characterized by categorical attributes. The output variable is expressed by a prediction interval in the rule consequent.

In conclusion, as shown by experimental results, the produced fuzzy rule-based models have a simple structure and satisfactory prediction accuracy, despite the high complexity of the prediction problem. The achieved results in terms of prediction accuracy are quite encouraging, taking into account that the few training samples used in this study do not provide a good coverage of the problem domain.

6. Conclusions

In this paper we have described a KDD methodology to construct fuzzy representations of knowledge from numerical data. The core of the proposed strategy consists in automatically capturing the unknown dependencies among data by adaptively clustering the input space, so as to obtain an initial fuzzy rule base that is subsequently refined to improve its accuracy. This knowledge extraction process is arranged into a neuro-fuzzy framework that performs a profitable integration between the neural network learning

(1) If Cu is in [240, 540], V is in [0, 1100], DRAWING is in {Source1, Source2, Source3},
Then V is in [0, 711];
(2) If Cu is in [220, 350], V is in [1700, 3400], DRAWING is in {Source2, Source3, Source4, Source5},
Then V is in [2720, 3890];
(3) If Cu is in [260, 540], V is in [1600, 3300], DRAWING is in {Source2, Source3},
Then V is in [1640, 2800];
(4) If Cu is in [0, 210], V is in [1100, 2900], DRAWING is in {Source1, Source2},
Then V is in [1140, 2310];
(5) If Cu is in [0, 260], V is in [0, 1400], DRAWING is in {Source2, Source3, Source4, Source5},
Then V is in [0, 877];
(6) If Cu is in [290, 590], V is in [950, 1710], DRAWING is in {Source4, Source5},
Then V is in [1590, 2750];
(7) If Cu is in [0, 290], V is in [0, 1400], DRAWING is in {Source1, Source2, Source3},
Then V is in [0, 690].

Fig. 6. The obtained fuzzy rule base related to the output variable Vanadium.

and the powerful knowledge representation of fuzzy rule-base models. In contrast to most existing fuzzy modeling approaches, no a priori knowledge is assumed on the process underlying the data: the only required information is represented by a finite set of input/output observations. The proposed framework provides for two further steps, intended to enhance the reliability and the comprehensibility of the extracted knowledge. On one side, a pre-processing stage performing variable selection is applied to simplify the task we are dealing with, thus improving both accuracy and readability of the obtained models. On the other side, a post-processing phase realizes an output granulation to estimate an uncertainty measure of the final results. As a case study, the proposed framework has been applied to a real-world prediction problem, involving prediction of chemical compounds present in ashes resulting from industrial combustion processes. This study confirms the effectiveness of the proposed KDD methodology and it contributes to the understanding of the possibilities to extract and represent knowledge about the toxicity of ashes derived from industrial combustion processes.

Acknowledgements

The authors would like to thank Mr. M. Giovannini in ENEL Produzione Ricerca, S.p.A., for his constructive suggestions and for providing the dataset.

References

[1] A. Abraham, Neuro-fuzzy systems: state-of-the-art modeling techniques, in: J. Mira, A. Prieto (Eds.), Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence, Springer, Berlin, 2001, pp. 269–276.
[2] L. Arafeh, H. Singh, S.K. Putatunda, A neuro-fuzzy logic approach to material processing, IEEE Trans. Systems Man Cybernet. 29 (1999) 362–370.
[3] R. Babuska, J.A. Roubos, H.B. Verbruggen, Identification of MIMO systems by input–output TS fuzzy models, in: Proc. IEEE Internat. Conf. on Fuzzy Systems (FUZZ-IEEE 1998), Anchorage, Alaska, 1998, pp. 657–662.
[4] R. Bellman, Adaptive Control Process: A Guided Tour, Princeton University Press, Princeton, NJ, 1961.
[5] C.M. Bishop, Neural Network for Pattern Recognition, Clarendon Press, Oxford, 1995.
[6] L. Caponetti, G. Castellano, A.M. Fanelli, A neuro-fuzzy system for document image segmentation and region classification, in: Proc. IEEE Internat. Workshop on Intelligent Signal Processing (WISP 2001), Budapest, Hungary, 2001, pp. 27–32.

[7] G. Castellano, A.M. Fanelli, A self-organizing neural fuzzy inference network, in: Proc. IEEE Internat. Joint Conference on Neural Networks (IJCNN 2000), Como, Italy, 2000, pp. 14–19.
[8] G. Castellano, A.M. Fanelli, Fuzzy inference and rule extraction using a neural network, Neural Network World J. 3 (2000) 361–371.
[9] G. Castellano, A.M. Fanelli, T. Roselli, Fuzzy rule extraction by a KBN approach, in: Proc. First Internat. ICSC Conf. on Neuro-Fuzzy Technologies (NF 2002), Havana, Cuba, 2002.
[10] G. Castellano, C. Castiello, A.M. Fanelli, M. Giovannini, A neuro-fuzzy system for ash property prediction, in: Proc. Second Internat. Workshop on Intelligent Systems Design and Applications (ISDA2002), Atlanta, USA, 2002, pp. 175–180.
[11] C. Castiello, G. Castellano, A.M. Fanelli, Classifying image pixels by a neuro-fuzzy approach, in: Proc. 10th IFSA World Congress (IFSA 2003), Istanbul, Turkey, 2003, pp. 253–256.
[12] J.Q. Chen, Y.G. Xi, Nonlinear system modeling by competitive learning and adaptive fuzzy inference system, IEEE Trans. Systems Man Cybernet. 28 (2) (1998) 231–238.
[13] S.L. Chiu, Selecting input variables for fuzzy models, J. Intelligent Fuzzy Syst. 4 (1996) 243–256.
[14] K. Cios, W. Pedrycz, R. Swiniarski, Data Mining Methods for Knowledge Discovery, Kluwer Academic Publishers, Dordrecht, 1998.
[15] I. Cloete, J.M. Zurada, Knowledge-based Neurocomputing, MIT Press, Cambridge, MA, 2000.
[16] W.A. Farag, V.H. Quintana, G. Lambert-Torres, A genetic-based neuro-fuzzy approach for modeling and control of dynamical systems, IEEE Trans. Neural Networks 9 (5) (1998) 756–767.
[17] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smith, R. Uthurusamy, Advances in Knowledge Discovery and Data Mining, MIT Press, Cambridge, MA, 1996.
[18] A. Gaweda, J. Zurada, Input selection in data-driven fuzzy modeling, in: Proc. 10th IEEE Internat. Conf. Fuzzy Systems, Melbourne, Australia, 2001.
[19] A. Gegov, Multilayer fuzzy control of multivariable systems by active decomposition, Internat. J. Intelligent Systems 12 (1997) 403–411.
[20] S. Geman, E. Bienenstock, R. Doursat, Neural networks and the bias/variance dilemma, Neural Computation 4 (1992) 1–58.
[21] P.A. Gili, M. Battipede, A comparative design of an MIMO neural adaptive rate damping for an nonlinear helicopter model, in: Proc. European Symp. on Artificial Neural Networks (ESANN 2000), Bruges, Belgium, 2000, pp. 159–164.
[22] S.K. Halgamuge, M. Glesner, Neural networks in designing fuzzy sys. for real world applications, Fuzzy Sets and Systems 65 (1) (1994) 1–12.
[23] M. Hanss, On developing enhanced fuzzy models for nonlinear process control, in: Proc. 17th NAFIPS Internat. Conf. (NAFIPS 1998), Pensacola Beach, FL, USA, 1998, pp. 365–369.
[24] S. Himavathi, B. Umamaheswari, A multi-step approach for modeling MIMO systems from input–output data, in: Proc. 27th Annu Conf. of the IEEE Industrial Electronics Society (IECON 2001), 2001, pp. 664–669.
[25] J.R. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Systems Man Cybernet. 23 (1993) 665–685.
[26] J.R. Jang, Input selection for ANFIS learning, in: Proc. IEEE Internat. Conf. Fuzzy Systems, New Orleans, USA, 1996.
[27] J. Kim, N. Kasabov, HyFIS: adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems, Neural Networks 12 (1999) 1301–1319.
[28] I.M. Kouatli, A simplified fuzzy multivariable structure in a manufacturing environment, J. Intelligent Manufacturing 5 (1994) 365–387.
[29] C.T. Lin, C.S.G. Lee, Neural-network-based fuzzy logic control and decision systems, IEEE Trans. Comput. 40 (12) (1991) 1320–1336.
[30] D.A. Linkens, M.-Y. Chen, Input selection and partition validation for fuzzy modeling using neural network, Fuzzy Sets and Syst. 107 (1999) 299–308.
[31] Y.Z. Lu, M. He, C.-W. Xu, Fuzzy modeling and expert optimization control for industrial processes, IEEE Trans. Control Systems Technol. 5 (1997) 2–12.
[32] A. Luk, S. Lien, Rival rewarded and randomly rewarded rival competitive learning, in: Proc. IEEE Internat. Joint Conf. Neural Networks, Washington, USA, 1999.
[33] T.M. Mitchell, Machine Learning, McGraw-Hill International Editions, New York, 1997.

[34] S. Mollov, T. van den Boom, F. Cuesta, A. Ollero, R. Babuska, Robust stability constraints for fuzzy model predictive control, IEEE Trans. Fuzzy Syst. 22 (1) (2002) 50–56.
[35] D. Nauck, F. Klawon, R. Kruse, Foundations of Neuro-Fuzzy Systems, Wiley, Chichester, 1997.
[36] D. Nauck, R. Kruse, NEFCLASS—A neuro-fuzzy approach for the classification of data, in: K.M. George et al. (Eds.), Applied Computing 1995, Proc. ACM Symp. on Applied Computing, Nashville, ACM Press, New York, 1995.
[37] J. Neter, W. Wasserman, M.H. Kutner, Applied Linear Statistical Models: Regression Analysis of Variance and Experimental Designs, Irwin, Homewood, IL, 1985.
[38] H. Nomura, Y. Hayashi, N. Wakami, A learning method of fuzzy inference rules by descent method, in: Proc. IEEE Internat. Conf. on Fuzzy Systems, CA, USA, 1992, pp. 203–210.
[39] Y. Shi, M. Mizumoto, A new approach of neuro-fuzzy learning algorithm for tuning fuzzy rules, Fuzzy Sets and Syst. 112 (1) (2000) 99–116.
[40] P. Stec, Y. Zhu, Some study on identification of ill-conditioned process for control, in: Proc. American Control Conference, Arlington, USA, 2001.
[41] M. Sugeno, T. Yasukawa, A fuzzy-logic-based approach to qualitative modeling, IEEE Trans. Fuzzy Syst. 1 (1993) 7–31.
[42] C.T. Sun, Rule-base structure identification in an adaptive-network-based fuzzy inference system, IEEE Trans. Fuzzy Syst. 2 (1) (1994) 64–73.
[43] H. Timm, Fuzzy cluster analysis of classified data, in: Proc. of Joint 9th IFSA World Congress and 20th NAFIPS Internat. Conf. (IFSA-NAFIPS 2001), Vancouver, Canada, 2001.
[44] L.X. Wang, Modeling and control of hierarchical systems with fuzzy systems, Automatica 33 (6) (1997) 1041–1053.
[45] L.X. Wang, J. Mendel, Generating fuzzy rules by learning from examples, IEEE Trans. Systems Man Cybernet. 22 (1992) 1414–1427.
[46] L. Xu, A. Krizak, E. Oja, Rival penalized competitive learning for clustering analysis, IEEE Trans. Neural Networks 4 (4) (1993) 636–649.
[47] X.-J. Zeng, M.G. Singh, Approximation theory of fuzzy systems—MIMO case, IEEE Trans. Fuzzy Syst. 3 (2) (1995) 219–235.