From knowledge-based to data-driven fuzzy modeling

Development, criticism, and alternative directions

Eyke Hüllermeier

Introduction

Since their inception 50 years ago, marked by Lotfi Zadeh's seminal paper [29], and rapid emergence in the following decades, fuzzy sets and fuzzy logic have found their way into numerous fields of application, such as engineering and control, operations research and optimization, databases and information retrieval, data analysis and statistics, just to name a few.

While different tools from fuzzy logic and fuzzy set theory (FST) have been employed in all these fields, it is fair to say that *fuzzy rule models* or *fuzzy rule-based systems* (FRBS) have received special attention. Indeed, rule-based models have always been a cornerstone of fuzzy systems and a central aspect of research in fuzzy logic – the term "fuzzy system" is mostly even used as a synonym for fuzzy rule-based system. To a large extent, the popularity of rule-based models can be attributed to their potential comprehensibility, a distinguishing feature and key advantage in comparison to "black-box" models such as neural networks.

Fuzzy systems provide an interface between humans and machines: Mathematical concepts such as fuzzy sets and generalized logical operators allow for an adequate formalization of vague cognitive concepts and linguistic expressions such as "high temperature." Moreover, they provide suitable means for reasoning with such concepts in a meaningful way. In principle, machines and computers thus become amenable to human expert knowledge [19]. The importance of rule-based methods can be explained by the fact that human experts often find it convenient to describe their knowledge in terms of IF–THEN rules, which typically connect the values of a set of (independent) input variables and those of one or more (dependent) output variables.

While corresponding aspects of knowledge representation and reasoning have dominated research in fuzzy logic for a long time, problems of *automated learning and knowledge acquisition* have more and more come to the fore in recent years [10]. There are several reasons for this development, notably the following. First, caused by the awareness of the well-known "knowledge acquisition bottleneck" and the experience that a purely knowledge-based approach to systems design is difficult, intricate, and tedious most of the time, there has been an internal shift within fuzzy systems research from *modeling* to *learning* and *adaptation*, i. e., from the knowledge-based to the data-driven design of fuzzy systems [2]. In fact, the latter not only suggests itself in applications where data is readily available, but can sometimes even be essential. In learning on data streams, for example, models are not only constructed once (from a static "batch" of data) but need to be updated continuously in an online manner [1], which cannot be accomplished by a human expert. Second, this trend has been further amplified by the great interest that the field of knowledge discovery in databases and its core methodological components, machine learning and data mining, have attracted in recent years [7]. Learning from data and data analytics have become ubiquitous topics in the era of "big data."

DOI 10.1007/s00287-015-0931-8

Eyke Hüllermeier
Department of Computer Science, University of Paderborn, Paderborn, Germany
E-Mail: eyke@upb.de

Abstract

This paper elaborates on a development in (applied) fuzzy logic that has taken place in the last couple of decades, namely, the complementation or even replacement of the traditional knowledge-based approach to fuzzy rule-based systems design by a data-driven one. It is argued that the classical rule-based modeling paradigm is actually more amenable to the knowledge-based approach, for which it has originally been conceived, while being less apt to data-driven model design. An important reason that prevents fuzzy (rule-based) systems from being leveraged in large-scale applications is the flat structure of rule bases, along with the local nature of fuzzy rules and their limited ability to express complex dependencies between variables. As an alternative approach to fuzzy systems modeling, we advocate so-called fuzzy pattern trees. Due to its hierarchical, modular structure and the use of different types of (nonlinear) aggregation operators, a fuzzy pattern tree has the ability to represent functional dependencies in a more flexible and more compact way.

This paper starts with a critical consideration of the transition from knowledge-based to data-driven fuzzy modeling. The basic claim we make is that fuzzy (rule-based) systems, which were originally introduced for the knowledge-based design of relatively small or at best moderately sized models, are not very amenable to the data-driven approach to model construction: they are inappropriate for large-scale modeling and tend to be inferior to mainstream machine learning methodology for model induction and adaptation. In our opinion, the fuzzy rule-based approach is adopted in a too uncritical way and is not sufficiently questioned in the research community. Indeed, apart from several indisputable advantages, this approach has a number of potential disadvantages. Two reasons for our reservation will be detailed in the paper:

- First, because of their flat structure, standard fuzzy (rule-based) systems are not scalable and hardly able to capture complex dependencies with many input variables, which are typical of present data-intense applications.
- Second, despite the opposing claims of fuzzy logic scholars, we argue that fuzzy systems automatically extracted from data easily lose one of their main advantages, namely, their interpretability and cognitive plausibility.

In the second part of the paper, we advocate an alternative approach to fuzzy systems modeling called *fuzzy pattern trees* [9, 23].[1] This approach is largely motivated by the disadvantages of rule-based system architectures. Because of its hierarchical, modular structure and the use of different types of (nonlinear) aggregation operators, a fuzzy pattern tree (FPT) has the ability to represent functional dependencies in a more flexible and more compact way, thereby offering a reasonable balance between accuracy and model transparency. In addition to giving an overview of the main modeling concepts, we address the problems of data-driven model calibration and FPT structure learning.

Knowledge-based versus data-driven fuzzy modeling

Knowledge-based modeling

The knowledge-based approach is at the origin of fuzzy rule-based systems and closely connected to the classical expert systems paradigm: A human expert seeks to formalize her knowledge about the relationship between certain variables of interest (for example, a control function mapping system states to control actions) using IF–THEN rules, taking advantage of fuzzy sets as a convenient interface between a qualitative, symbolic and a quantitative, numerical level of knowledge representation. As an illustration, consider a recent application from the textile industry, namely, modeling color yield (K/S) in polyester high-temperature dyeing as a function of disperse dye concentration (`conc`), temperature (`temp`), and time. The human expert describes this dependency using the following rules [17]:

1. If `temp` is low and `time` is low and `conc` is low, then K/S is very low.
2. If `temp` is medium and `conc` is high, then K/S is high.
3. If `temp` is high and `conc` is low, then K/S is medium.

[1] In terms of its content, this part largely overlaps with [21].

4. If `temp` is low and `time` is high and `conc` is low, then K/S is very low.
5. If `temp` is high and `conc` is high, then K/S is very high.
6. If `temp` is medium and `time` is low and `conc` is high, then K/S is medium.
7. If `temp` is medium and `time` is high and `conc` is high, then K/S is high.
8. If `temp` is low and `time` is low and `conc` is high, then K/S is low.

A set of informal rules of that kind is then translated into a mathematical model M by

- assigning a fuzzy subset to each linguistic term (such as "high temperature"), and thereby a fuzzy partition for each variable,
- choosing a generalized logical operator for conjunction ("and") and implication ("then"),
- specifying an inference procedure such as Mamdani-Assilian [15] or Takagi-Sugeno inference [25],
- defining a fuzzification and a defuzzification procedure (mapping, respectively, nonfuzzy to fuzzy and fuzzy to nonfuzzy quantities).

As can be seen, a fuzzy model M of that kind is highly "parameterized" and involves many degrees of freedom: the structure of the rules, the parameters of fuzzy sets, etc. Eventually, the system as a whole (cf. Fig. 1) realizes a mapping

$$f_M : \mathbb{R}^d \longrightarrow \mathbb{R} \ .$$

In the case of the above example, this function would simply map each input triple $(conc, temp, time) \in \mathbb{R}^3$ to a number $K/S \in \mathbb{R}$.

Specifying a fuzzy system in this way is often a difficult and tedious task. Even if the expert is able to communicate her knowledge in an appropriate form, and willing to accept standard choices for fuzzification and defuzzification as well as logical operators and fuzzy inference, the specification of a complete and sufficiently consistent knowledge base, consisting of the rules and fuzzy sets, is quite demanding. In particular, the number of rules may become very large. In our above example, this number is still rather small because the number of input attributes is limited to only three. In general, however, the number of rules quickly grows with the number of attributes – a point that we shall come back to below. Finally, "tuning" the system in case it does not immediately realize the desired input/output relationship is difficult, too, because the different components of the system (fuzzy sets, rules, logical operators, (de-)fuzzification procedures) are interacting in a complicated and highly nonlinear way.

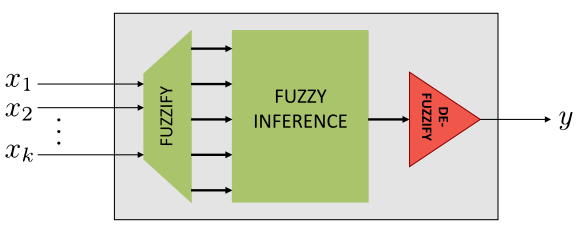

Fig. 1 Structure of a fuzzy (rule-based) system

Data-driven modeling

Given these difficulties, it is hardly surprising that data-driven approaches to fuzzy systems modeling have been considered as an alternative. In fact, this alternative suggests itself in case data about the process to be modeled is available. Imagine, for instance, that a set of N measurements

$$(x_n, y_n) = (conc_n, temp_n, time_n, K/S_n) \in \mathbb{R}^3 \times \mathbb{R}$$

is available in our above example. One could then be tempted to "fit" a fuzzy model M to this data, that is, to search for an instantiation of the model components such that the induced mapping f_M reproduces the data sufficiently well (i. e., $f_M(x_n) \approx y_n$ for $n = 1, \ldots, N$).

Obviously, this process of reverse-engineering a fuzzy system bears a close resemblance to standard statistical regression analysis. Seen from this perspective, a fuzzy system can simply be considered as a (nonparametric) regression function. This view becomes especially apparent for certain types of fuzzy systems, for example, systems with Gaussian fuzzy sets and Takagi-Sugeno inference, which are formally more or less equivalent to common regression techniques such as radial basis function (RBF) networks.

Fuzzy systems are of course not limited to the representation of regression functions but can also be used for classification, that is, for implementing functions with a categorical output. In this case, the consequent of single rules is usually a class as-

signment, i. e., a singleton fuzzy set.[2] Evaluating a rule base (à la Mamdani-Assilian) thus becomes trivial and simply amounts to "maximum matching," that is, searching the maximally supporting rule for each class. Thus, much of the appealing interpolation and approximation properties of fuzzy inference gets lost, and fuzziness only means that rules can be activated to a certain degree. There are, however, alternative methods that combine the predictions of several rules into a classification of the query [4].

A fuzzy model consists of two types of components, a qualitative one that determines the structure of the model and essentially corresponds to the (linguistic) rules, and a quantitative one that comprises all numerical parameters of fuzzy sets, logical operators, etc. A plethora of strategies have been developed for determining these components in a data-driven way. Especially important in this regard are hybrid methods that combine fuzzy logic with other "soft computing" methodologies, notably evolutionary algorithms and neural networks. For example, evolutionary algorithms are often used in order to optimize ("tune") a fuzzy rule base or for searching the space of potential rule bases in a (more or less) systematic way [5]. The combination of fuzzy systems with neural networks leads to so-called *neuro-fuzzy* systems [12, 18].

Combined knowledge-based and data-driven modeling

In addition to a purely knowledge-based and a purely data-driven approach, there is of course also the possibility to combine these two. For example, a human expert may specify the qualitative part of a fuzzy model by providing a set of linguistic rules, whereas the quantitative part is determined automatically by an optimization method that fits the structure to a given set of data. In other words, the data is used to "calibrate" the structure of the model as specified by the expert. In the context of the colour yield application mentioned above, this approach has been used successfully in [16].

From a (machine) learning point of view, the part of the model that is pre-defined by the expert serves as a restriction of the underlying model space, i. e., the set of candidate models the learning algorithms can choose from (or, using machine learning terminology, it incorporates an *inductive bias*). A restriction of that kind can be very useful, as it may help the algorithm to learn more quickly and avoid the risk of overfitting the data. Conversely, if not completely correct, the bias incorporated by the expert may also prevent the algorithm from finding the truly best model. In general, the importance of incorporating expert knowledge increases with the sparsity of the data. If data abounds, it can compensate for the expert knowledge, and purely data-driven approaches are often superior to knowledge-based or hybrid alternatives, at least in terms of accuracy.

Limitations of fuzzy rule systems

With the ever increasing availability of data, fostered by technological advances in data acquisition, storage and management, the data-driven approach to systems design has become more and more prevalent in the previous years, not only in the field of fuzzy logic but in artificial intelligence (AI) in general – this is why (inductive) machine learning methodology has gained in importance as compared to more classical AI topics of knowledge representation and (deductive) reasoning.

Given the original motivation of fuzzy modeling as a means for expressing expert knowledge, one may wonder whether this methodology is equally applicable to the data-driven paradigm and, not less importantly, to what extent it may usefully complement the arsenal of alternative statistical and machine learning methods, including neural networks, support vector machines, and decision trees, amongst others. This question especially arises since standard machine learning methods are amenable to specific optimization techniques (such as backpropagation in the case of neural networks or quadratic programming in the case of support vector machines), and therefore much more efficient from an algorithmic point of view. Indeed, the development of a machine learning method normally involves algorithmic and computational considerations from the very beginning. In the case of fuzzy systems, these aspects became relevant only later on, and because of their complex structure, the only way to identify fuzzy models is via general-purpose optimization tools like the "big hammer" of evolutionary algorithms.

[2] More generally, a rule consequent can suggest different classes with different degrees of certainty.

Since we have been discussing the role of fuzzy logic in machine learning on a quite general level elsewhere [10, 11], this topic will not be deepened any further in this article. Instead, because our focus here is on fuzzy systems in the sense of fuzzy rule models, the remainder of this section is devoted to a brief discussion of two issues directly related to the use of such systems for learning and data-driven model construction.

The problem of a flat structure

The color yield example above is a very simple, low-dimensional model with three inputs and one output variable. In spite of its practical relevance, most applications will typically involve much more inputs – in modern, data-intense applications, it is not uncommon to deal with hundreds or thousands of variables. But even if the number of variables goes beyond a handful, rule-based systems may become problematic [13]. A major reason that prevents such systems from scaling to more complex applications is their flat structure.

Rules are purely local entities that only cover a small portion of the data space (in the form of more or less axis-parallel rectangles) and that are not able to flexibly exploit specific dependencies or independencies between individual variables. Therefore, a large number of such rules is typically needed in order to describe a global relationship between input and output variables. Regardless of whether grid-based methods (starting from fuzzy sets on the individual dimensions and defining rules as products of these sets) or covering techniques (starting from multi-dimensional rules/clusters in the input space and defining fuzzy sets as one-dimensional projections) are used, the number of rules will grow exponentially or at least almost exponentially with the dimensionality of the input space.

Successful tools for large-scale modeling and learning, such as graphical models and deep neural networks, distinguish themselves by the ability to capture (partial) independencies between variables and/or by conquering complexity through abstraction and hierarchical structuring. This is in contrast to the flat, grid-like structure of standard (fuzzy) rule bases.

Since a fuzzy system eventually represents a (real-valued) function, it is of course possible to combine fuzzy rule-based modeling with generic hierarchical decomposition techniques. For example, suppose a function $f(\cdot)$ with three input variables can be expressed in the following form:

$$f(x, y, z) = g(x, h(y, z))$$

Obviously, both $g(\cdot)$ and $h(\cdot)$ can then be expressed in the form of fuzzy rule bases, giving rise to two fuzzy systems with two-dimensional input space instead of one such system with three inputs. In fact, what is thus obtained is a specific type of *hierarchical fuzzy system* – other types have been proposed in the literature, see [26] for an overview.

Although hierarchical modeling techniques may to some extent overcome or at least alleviate the problems mentioned above, it seems that hierarchical fuzzy systems have their own drawbacks and have not been widely adopted in practice so far. In fact, designing such systems, whether in a knowledge-based or a data-driven way, is not an easy task. Moreover, there is a risk of further compromising the interpretability of fuzzy systems, which, as will be argued in the next section, is an issue already in the case of standard (flat) fuzzy rule bases.

Another type of hierarchical fuzzy system, called *fuzzy pattern trees*, will be discussed in Section 4 below. In contrast to rule-based systems, pattern trees exhibit an *inherently* hierarchical structure. Moreover, each "inner node" of the hierarchy is realized in the form of a simple (and easily interpretable) aggregation function (instead of a set of rules wrapped in a fuzzification and defuzzification procedure).

The myth of interpretability

Interpretability is one of the core arguments often put forward by fuzzy scholars in favor of fuzzy models, regardless of whether these models have been constructed in a knowledge-based or a data-driven way. Unfortunately, this argument is not well supported from a scientific point of view, which is partly due to the difficulty of measuring interpretability and model transparency in an objective manner. Instead, the claim that rules are more understandable than "formulas" appears to be a commonplace one that is simply taken for granted. Without denying the potential usefulness of fuzzy logic in constructing interpretable models in general, we are convinced that current methods for data-driven construction of fuzzy models produce results that are not at all more interpretable than any

other types of model, and often even less. There are several reasons for this scepticism.

A first problem is connected to the previous discussion and concerns the size of practically relevant models. Even if a single rule might be understandable, a rule-based model with a certain level of accuracy will typically consist of many such rules, which may interact in a nontrivial way and might be hard to digest as a whole. On top of this, fuzzy models often allow for rule weighing and may involve complicated inference schemes for aggregating the outputs of individual rules into an overall prediction. Of course, the problem of complexity is shared by other, nonfuzzy methods, too. For example, decision trees are often praised as being highly interpretable, also in mainstream machine learning. This might indeed be true as long as trees are sufficiently small. In real applications, however, accurate trees are often large, comprising hundred of nodes. Again, interpretability is highly compromised then.

When fuzzy sets are constructed in a data-driven way, it is not at all clear that these sets can be associated with meaningful linguistic labels – let alone labels the semantic interpretation of which will be shared among different users. In fact, one should realize that the fuzzy sets produced are strongly influenced by the data set, which is a random sample, and fuzzy partitions might be strongly influenced by outliers in the data. Moreover, fuzzy sets are in the first place tuned toward good approximation and accurate predictions of the function f_M implemented by the fuzzy model, and not toward meaningful semantics.

Likewise, it is rarely discussed in which way a model is eventually presented to the user. Are the fuzzy sets specified in terms of their membership functions, in which case the user might be overloaded with technical details, or is it just presented in a linguistic form? As mentioned before, the latter presupposes an appropriate assignment of linguistic labels to fuzzy sets, which is difficult to establish.

For these reasons, the (alleged) interpretability of fuzzy systems cannot simply be transferred from knowledge-based to the case of data-driven fuzzy modeling. In fact, as an important difference between these approaches, one should notice that, in the data-driven approach, the human is no longer at the origin of the model but changes her role from the "producer" to the "consumer" of a model. Obviously, a model constructed by the expert herself is very close to what she has in mind. Besides, models of that kind are typically small, comprising only a few input variables. Therefore, one can indeed suppose such models to be understandable. These properties are no longer valid in the case of a data-driven approach, however.

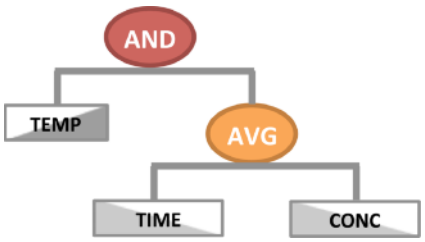

Fig. 2 Example of an FPT modeling colour yield as a function of concentration, temperature and time

Fuzzy pattern trees

In this section, the model class of fuzzy pattern trees [9] is advocated as an interesting alternative to conventional fuzzy rule models. A fuzzy pattern tree (FPT) is a modular, hierarchical structure and, moreover, disposes of a wide spectrum of generalized aggregation operators. In a sense, it can be seen as a generalisation of standard and/or trees.

The characteristic property of the hierarchical approach, namely the (recursive) partitioning of a problem into simpler subproblems, with a subsequent combination of the corresponding solutions, appears to be a key prerequisite for the controllability of complex systems. Besides, it allows for the consideration of a system on different levels of abstraction.

Figure 2 shows an example of an FPT for our previous example, modeling color yield as a function of concentration, temperature, and time. The model specifies fuzzy conditions for "high color yield" and suggests that the overall yield is a conjunction of two such conditions: a high temperature and a second criterion. The second criterion is a complex criterion which is again decomposed into two subcriteria (low time and concentration), which are combined by an average. Figure 3 shows another example, namely, an FPT modeling the quality of red wine depending on its chemical properties. Both models have been constructed in a data-driven way, using the techniques to be outlined below.[3]

[3] The wine quality data, which contains ten input attributes, can be found at https://archive.ics.uci.edu/ml/datasets/Wine+Quality.

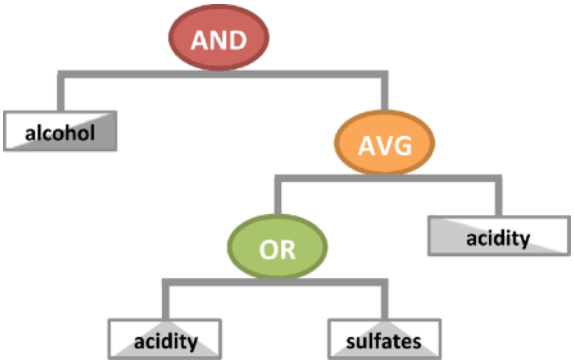

Fig. 3 Example of an FPT modeling the quality of red wine

In the following, the model class of fuzzy pattern trees will be described in more detail. Then, the data-driven design of FPTs will be addressed, which can be restricted to the calibration of model parameters or even comprise the identification of the model structure. For technical details of fuzzy pattern tree learning, we refer the reader to [23, 24].

Structure of fuzzy pattern trees

An FPT realizes a mapping from an input space $\boldsymbol{X} = \boldsymbol{X}_1 \times \cdots \times \boldsymbol{X}_m$ to an output space $\boldsymbol{Y}$. Thus, inputs are vectors $x = (x_1, \ldots, x_m)$, the components x_i of which assume values in $\boldsymbol{X}_i$; for simplicity, we will subsequently assume that all domains $\boldsymbol{X}_i$ are numerical (i. e., subsets of $\mathbb{R}$). As for the output, we shall consider two cases, namely the binary one where $\boldsymbol{Y} = \{0, 1\}$, and the numerical one with $\boldsymbol{Y} = \mathbb{R}$ or $\boldsymbol{Y} = [a, b] \subset \mathbb{R}$.

An FPT is a binary tree having the following properties:

- Each inner node N_i is associated with a (binary) aggregation operator $A_i : [0, 1]^2 \to [0, 1]$.
- The root node N_0 is additionally associated with a function $\nu : [0, 1] \to \boldsymbol{Y}$ (defuzzifier).
- Each leaf node L_j is associated with a membership function $\mu_j : \boldsymbol{X}_{\nu(j)} \to [0, 1]$ (fuzzifier).

Given a query input $x = (x_1, \ldots, x_m) \in \boldsymbol{X}$, an FPT is evaluated in a recursive way according to the following rules:

- The root computes $\nu(A_0(y_l, y_r))$ and thus defines the output of the tree; y_l and y_r are, respectively, the outputs produced by the left and right child node.
- Each inner node N_i computes $A_i(y_l, y_r)$ as an output, where y_l and y_r are again the outputs of the left and right child node, respectively.
- Each leaf node computes $\mu_j(x_{\nu(j)})$.

Thus, in an FPT, the inputs x_i are propagated from the bottom (leaf nodes) to the top (root). Combining the μ_j and the A_i into a single "fuzzifier" μ and an aggregation A, respectively, the mapping defined by an FPT can be written in a somewhat sloppy way as $\nu(A(\mu(x)))$. In the following, we shall discuss the components of an FPT in a little more detail.

Fuzzification and defuzzification. Each domain $\boldsymbol{X}_i$ is discretized by means of a fuzzy partition, and the fuzzy sets are marked with linguistic terms such as 'large', 'medium,' or 'small' (making sure that these terms are meaningful descriptions). The definition of the function $\nu : [0, 1] \to \boldsymbol{Y}$ and its inverse $\nu^{-1} : \boldsymbol{Y} \to [0, 1]$ depend on $\boldsymbol{Y}$. For $\boldsymbol{Y} = [a, b]$, one may simply let $\nu^{-1}(z) = (z - a)/(b - a)$. Like in the case of neural networks, the case $\boldsymbol{Y} = \mathbb{R}$ suggests a sigmoid transformation: $\nu^{-1}(z) = \big(1 + \exp(-\alpha \cdot z)\big)^{-1}$. In the case of a binary output, we simply set $\nu(z) = 1$ if $z > \theta$ and $\nu(z) = 0$ otherwise, where $0 < \theta < 1$ is a threshold (ν^{-1} is then the identity by definition).

Aggregation functions. Aggregation functions are functions with specific properties, such as monotonicity and commutativity; in general, they are defined for two arguments, but thanks to their associativity, they can be extended to functions of arity n in a canonical way. In the literature, one distinguishes three types of aggregation functions: conjunctive, compensatory, and disjunctive [8]. When defining an order relation $\leq$ on aggregation functions in agreement with the standard (pointwise) order on functions, then conjunctive aggregations A are those with $A \leq \min$, compensatory (averaging) those with $\min \leq A \leq \max$, and disjunctive those for which $\max \leq A$.

If $A \leq A'$ for two aggregations A and A', then A can be said to be "more strict" than A': Given the same inputs (e. g., the degree of fulfillment of two criteria), A' will always produce a value at least as high as the one produced by A. The conjunctive and disjunctive aggregation generalizes, respectively, the logical AND and OR, whence the former is more strict than the latter. The averages are located in-between these two classes. They produce

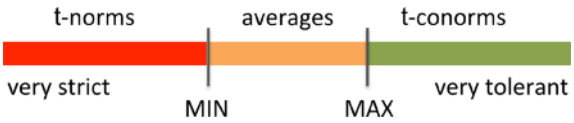

Fig. 4 Spectrum of aggregation functions

a compromise between the input values and fill the "gap" between the minimum and the maximum (see Fig. 4).

As usual in fuzzy logic, we make use of the class of *triangular norms* (t-norms) for modeling the logical conjunction. T-norms are monotone, associative and commutative functions $[0,1]^2 \to [0,1]$ having 1 as a neutral element [14]. Correspondingly, the dual class of t-conorms (which is characterized by the same properties except having 0 instead of 1 as a neutral element) serves for modeling the logical disjunction. Especially appealing from a modeling point of view is the use of *parametrized* families of such norms (such as the Dubois-Prade family [6]), because they allow one to control the "strictness" of an aggregation by means of a single parameter. As an averaging operator, we make use of the so-called *Choquet integral*, which closes the gap between the minimum and the maximum [8] and covers the *weighted arithmetic mean* as well as the *ordered weighted average* (OWA) operator [27] as special cases.

Modeling and interpretation

The fuzzy sets at the leaf nodes of an FPT, applied to the respective variables, can be considered as (unary) predicates and perhaps expressed linguistically (e. g., "the level of alcohol is high"). Likewise, the membership function associated with the root defines a fuzzy predicate that we have considered in a generic way as "the outcome is high" so far. The inner part of an FPT defines a criterion for the fulfillment of this predicate, which is recursively composed of subcriteria. Eventually, an FPT thus specifies (fuzzy) conditions on the input variables that guarantee "a high output," or, more generally, that the predicate at the root node is satisfied to a high degree.

A human expert will typically design a model of this kind in a "top-down" manner, that is, by decomposing criteria into subcriteria in a recursive way. The advantages of such an approach are well-known in other fields, too; as an example, let us mention the *analytic hierarchy process* in decision support [20]. For each criterion, one has to determine the way in which the subcriteria ought to be combined (conjunctive, compensatory, disjunctive) and how the corresponding aggregations are parametrized (e. g., the strictness of a conjunction or the weights of the subcriteria in the arithmetic mean).

The hierarchical structure guarantees a certain modularity of the approach, which is also helpful for the interpretation of an FPT: Each criterion is "explained" by the respective subcriteria and the aggregation. Again, this explanation can be continued in a recursive way, completely independently of the rest of the model. Thus, not only the model itself can be analyzed, but also a concrete prediction: The numerical output of a node is explained by its inputs, i. e., the degree of fulfillment of the subcriteria, plus the aggregation of these inputs.

Comparison with rule-based models

In [22], an extension of the FPT approach is proposed in which a model is specified in terms of an ensemble of pattern trees. For instance, instead of using a single tree in our example, which specifies the conditions for a "high quality" (and immediately implies a low quality if these conditions are not satisfied), one may add corresponding trees for "medium quality" and "low quality". To make an overall prediction, it is then necessary to combine the outputs of these trees.

This approach is close to rule-based fuzzy systems: Each FPT can be associated with a rule or a set of rules with equal or similar consequent; the aggregation of the outputs of the trees then essentially corresponds to the step of defuzzification in rule systems. Yet, FPTs arguably have a number of advantages, notably the following.

A hierarchical instead of a flat structure often allows models to be represented in a more compact way. One reason is that, in the class of FPTs, much more transformations preserving equivalence are possible than in the class of rule models; for example, while the expression $\max\{\min\{A,B\},\min\{A,C\}\}$ can be considered as a disjunction of two rules with antecedents $A \wedge B$ and $A \wedge C$, respectively, the same is not true for the logically equivalent expression $\min\{A,\max\{B,C\}\}$. Moreover, the class of analytical expressions that can be represented, as well as the "degree of nonlinearity", are significantly increased thanks to the possibility of recursion. For example, with an FPT of depth k it is possible to model all

monomials of degree k by just using the simple product as a t-norm, even if all membership functions are linear.

In this regard, it is interesting to note that similar advantages of "deep" over "flat" structures have also been observed in other domains, currently for example in the field of *deep learning* [3]: Although it is true that neural networks with a single hidden layer exhibit universal approximation capabilities, the practical realization of this theoretical property may require an extremely large number of neurons in this layer. The same approximation quality might be achieved with a significantly smaller number of neurons if these are distributed on several layers and connected in a proper way. Likewise, the universal approximation property of fuzzy systems typically comes at the price of an excessively large number of rules.

As explained in Section 3.1, fuzzy rule-based models can be used within a hierarchical modeling scheme, too. Therefore, the above remarks need to be put in perspective. Nevertheless, labeling an inner node with a single aggregation function appears to be all the more simpler than describing each such aggregation in the form of a fuzzy rule-based system.

Indeed, the use of a larger class of aggregation functions, including generalized averaging operators, is another important advantage of FPTs. This is nicely exemplified by the weighted sum: While this operator allows one to describe linear or piecewise linear functions with a single node, the approximation of such functions with axis-parallel rules can become very complex.

One may argue that the use of different types of aggregation operators may compromise the interpretability of an FPT. In our opinion, however, this is not the case: Each operator itself is easily interpretable, and thanks to the modularity of an FPT, it can indeed be considered independently of the rest of the model.

Data-driven model construction

In the case of data-driven design, (possibly noisy) data about the input/output behavior of the system under consideration is supposedly available:

$$\mathcal{T} = \left\{(x_n, y_n)\right\}_{n=1}^{N} \subset \boldsymbol{X} \times \boldsymbol{Y} \ ,$$

where x_n is an observed input vector and y_n the corresponding output. These data can be used to replace or at least to complement the knowledge-driven model construction of an expert. Indeed, while human experts are often able to describe the qualitative structure of a model, including the type of aggregation (conjunctive, compensatory, disjunctive), they find it hard to parametrize the operators. An obvious idea, then, is to adapt the parameters to the data in an optimal way – a process we refer to as *model calibration*. Without going into technical detail, we refer to [28] for an approach to calibrating fuzzy pattern trees.

If no prior knowledge is available, one can try to extract both the model structure and the parameters from the data. This is a problem of model induction typical of machine learning and comparable, for example, to the learning of Bayesian networks, which also consist of a qualitative (the graph structure) and a quantitative part (the conditional probability distributions). In [23], a top-down approach to FTP induction is proposed. The core of our learning algorithm is a *beam search*, a heuristic search strategy that maintains a set of candidate solutions (FPTs in our case), and the problem of parameter estimation is embedded in this process. In [24], several extensions of the above algorithm are proposed that aim at making it faster without compromising predictive accuracy. These extensions include the use of adaptive sampling schemes as well as heuristics for guiding the growth of pattern trees. Again, a detailed exposition of such technical details is beyond the scope of this article.

Conclusion

Fuzzy modeling was originally conceived as a human/machine interface, namely, for translating informal rules involving linguistic terms with vague semantics into precise mathematical models amenable to computerized information processing. In the spirit of the classical expert systems paradigm, which dominated research in AI in the 1980s, fuzzy systems have been used quite successfully in many applications. More recently, however, the interest has shifted from the traditional knowledge-based approach to systems design to a data-driven one, that is, the automatic extraction of fuzzy models from data.

Although this development is a coherent reaction to the increased availability of data and completely in line with the growing importance of machine learning, we suspect that traditional fuzzy

systems lose most of their merits when being designed in a purely data-driven manner. Moreover, as a tool for model induction and predictive modeling, fuzzy approaches have a hard time competing with modern machine learning methodology, especially with regard to algorithmic aspects, computational efficiency, and theoretical foundations (guarantees for generalization performance, support of model selection, etc.). The scalability of fuzzy systems is severely hampered by the flat structure of standard rule bases, and even if this problem could in principle be mitigated by hierarchical variants, large-scale applications of whatever kind of fuzzy system are difficult to find. One of the key advantages of the traditional approach, namely, interpretability and model transparency, tends to become questionable for the data-driven approach. Moreover, a proper handling of fuzziness in knowledge representation is no longer needed, since data is normally precise, and the alleged fuzziness "learned" from the data is at best suspicious.

Needless to say, our criticism of the traditional rule-based paradigm should not be misunderstood as denying the usefulness of fuzzy logic in machine learning in general. For example, the model architecture of pattern trees, which we reviewed in the second part of the paper, is an interesting alternative with many appealing properties – the use of generalized (fuzzy) logical and averaging operators is one of the key features of this approach. For a broader discussion of how fuzzy logic can contribute to machine learning, we refer to [11].

References

1. Angelov P, Filev D, Kasabov N (2010) Evolving Intelligent Systems. John Wiley and Sons, New York
2. Babuska R (1998) Fuzzy Modeling for Control. Kluwer Academic Publishers, Boston
3. Bengio Y (2009) Learning deep architectures for AI. Foundations Trends Machine Learning 2(1):1–127
4. Cordon O, del Jesus MJ, Herrera F (1998) Analyzing the reasoning mechanisms in fuzzy rule based classification systems. Mathware Soft Comput 5:321–332
5. Cordon O, Gomide F, Herrera F, Hoffmann F, Magdalena L (2004) Ten years of genetic fuzzy systems: current framework and new trends. Fuzzy Set Syst 141(1):5–31
6. Dubois D, Prade H (1980) New results about properties and semantics of fuzzy set-theoretic operators. In: Wang PP, Chang SK (eds) Fuzzy Sets: Theory and Applications to Policy Analysis and Information Systems, Plenum Press, New York
7. Fayyad UM, Piatetsky-Shapiro G, Smyth P (1996) From data mining to knowledge discovery: an overview. In: Advances in Knowledge Discovery and Data Mining. MIT Press, pp 1–34
8. Grabisch M, Marichal JL, Mesiar R, Pap E (2009) Aggregation Functions. Cambridge University Press
9. Huang Z, Gedeon TD, Nikravesh M (2008) Pattern tree induction: a new machine learning method. IEEE T Fuzzy Syst 16(4):958–970
10. Hüllermeier E (2005) Fuzzy sets in machine learning and data mining: Status and prospects. Fuzzy Set Syst 156(3):387–406
11. Hüllermeier E (2011) Fuzzy machine learning and data mining. WIREs Data Min Knowl Disc 1(4):269–283
12. Jang JSR (1993) ANFIS: Adaptive-network-based fuzzy inference systems. IEEE T Syst Man Cyb 23:665–685, 1993
13. Jin Y (2000) Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement. IEEE T Fuzzy Syst 8(2):212–221
14. Klement EP, Mesiar R, Pap E (2002) Triangular Norms. Kluwer Academic Publishers
15. Mamdani E, Assilian S (1975) An experiment in linguistic synthesis with a fuzzy logic controller. Int J Man-Mach Stud 7:1–13
16. Nasiri M, Fober T, Senge R, Hüllermeier E (2013) Fuzzy pattern trees as an alternative to rule-based fuzzy systems: knowledge-driven, data-driven and hybrid modeling of color yield in polyester dyeing. In: Proceedings IFSA–2013, World Congress of the International Fuzzy Systems Association, Edmonton, Canada, pp 715–721, 2013
17. Nasiri M, Hüllermeier E, Senge R, Lughofer E (2011) Comparing methods for knowledge-driven and data-driven fuzzy modeling: A case study in textile industry. In: Proceedings IFSA–2011, World Congress of the International Fuzzy Systems Association, Surabaya and Bali Island, Indonesia, pp RW-103-1–6, 2011
18. Nauck D, Klawonn F, Kruse R (1997) Foundations of Neuro-Fuzzy Systems. John Wiley and Sons, Chichester, UK
19. Passino KM, Yurkovich S (1998) Fuzzy Control. Addison-Wesley
20. Saaty TL (1980) The Analytic Hierarchy Process. McGraw-Hill
21. Senge R, Fober T, Nasiri N, Hüllermeier E (2012) Fuzzy Pattern Trees: ein alternativer Ansatz zur Fuzzy-Modellierung. at – Atomatisierungstechnik 60(10):622–629
22. Senge R, Hüllermeier E (2010) Pattern trees for regression and fuzzy systems modeling. In: 2010 IEEE International Conference on Fuzzy Systems (FUZZ)
23. Senge R, Hüllermeier E (2011) Top-down induction of fuzzy pattern trees. IEEE T Fuzzy Syst 19(2):241–252
24. Senge R, Hüllermeier E (2015) Fast fuzzy pattern tree learning. IEEE T Fuzzy Syst PP(99):1
25. Takagi T, Sugeno M (1985) Fuzzy identification of systems and its applications to modeling and control. IEEE T Syst Man Cyb 15(1):116–132
26. Torra V (2002) A review on the construction of hierarchical fuzzy systems. Int J Intell Syst 17(5):531–543
27. Yager RR (1988) On ordered weighted averaging aggregation operators in multi criteria decision making. IEEE T Syst Man Cyb 18(1):183–190, 1988
28. Yi Y, Fober T, Hüllermeier E (2009) Fuzzy operator trees for modeling rating functions. Int J Comput Intell Appl 8(4):413–428
29. Zadeh LA (1965) Fuzzy sets. Inform Control 8(3):338–353