Fuzzy-System Kernel Machines
A Kernel Method Based on the Connections Between Fuzzy Inference Systems and Kernel Machines

Jorge Guevara, *IBM Research,* Jerry M. Mendel, *Life Fellow, IEEE,* and R. Hirata Jr., *University of Sao Paulo*

***Abstract*—This work introduces the *Fuzzy-System Kernel Machines* — a class of machine learning models based on the connection between fuzzy inference systems and kernel machines. For the connection, we observed a relationship between the Representer Theorem of kernel methods and the functional representation of non-singleton fuzzy systems (NSFSs). We found that the *non-singleton kernel on fuzzy sets* — a kernel defined in this work — is the core element allowing this two-way connection perspective. Consequently, a fuzzy system trained with the kernel method can be regarded as a kernel machine, and vice-versa a kernel machine trained with a *non-singleton kernel on fuzzy sets* can be interpreted as a fuzzy system. We conducted several experiments in supervised classification to understand the generalization power and properties of the proposed Fuzzy-Systems Kernel Machines.**

***Index Terms*—Kernel methods, kernel machines, fuzzy systems, non-singleton fuzzy systems.**

I. Introduction

Fuzzy systems and kernel machines are two important fields of scientific research that have greatly impacted not only science but several industrial applications. Fuzzy systems are learning machines based on fuzzy logic inference. Their learning process can be done using numerical data (data-driven) and/or using knowledge in the form of linguistic IF-THEN rules (knowledge-driven) [1]. On the other hand, the kernel method [2]–[4] is an approach for machine learning based on two parts: 1) a mapping defined by a kernel function that embeds the data into a feature space, and; 2) an algorithm defined on this feature space. Machine learning models based on the kernel method are known as kernel machines. Table I depicts a side-by-side comparison of relevant concepts between fuzzy systems and kernel methods. We explain each one of these concepts in more detail in Sections II and III.

TABLE I
General overview of fuzzy systems and kernel methods

	Fuzzy Inference Systems	Kernel Machines
Methods	Fuzzy sets and Fuzzy Logic	The kernel method
Input Space	Fuzzy sets	String, graphs, measures, sets, sequences, etc.
Structure	Fuzzifier, rules, inference engine, defuzzifier	Loss-penalty models with kernels
Learning	Tuning fuzzy sets parameters, number of rules, rule parameters, T-norm operators	Coefficient estimation based on the Representer Theorem
Function approximation	Universal approximator	Fuction approximation in kernel space
Main advantages	Linguistic and numerical information for function approximation	Modularity, no input constraints, Representer Theorem

In this paper, we provide the first step to research the connections between fuzzy systems and kernel machines, so as to discover how they both can benefit from each other. A direct consequence of this two-way connection is that a class of fuzzy systems can be interpreted as kernel machines and a class of kernel machines can be understood as being fuzzy inference systems. The connection will lead to the adoption of the main properties and advantages of both methods, in both directions[1]. Doing this, we conjecture the following benefits for fuzzy inference systems and kernel machines[2].

Benefits for fuzzy systems

- more resistance to the curse of dimensionality
- function approximation with explicit regularization
- Representer Theorem for fuzzy systems which implies the training of fuzzy systems by kernel algorithms

Benefits for kernel machines

- design of rule-base kernel machines
- explainability of black-box kernel machines
- attribute noise-resistant kernel machines

As the forest of fuzzy systems is large [1], we limit this work to the study of the connection between *non-singleton* Mamdani fuzzy systems (NSFS) and regularized kernel machines [4], [6][3]. Table II shows two promising novel machine learning models — that we called *Fuzzy-System Kernel machines* (FSKM) — based on the studied connections. The first studied connection produces the model FSKM-k which is both a fuzzy system and a kernel machine. The second researched connection provides the model FSKM-ϕ — a kernel machine expressing the sum of two fuzzy systems. We tested FSKM-k and FSKM-ϕ models in a set of supervised classification experiments to show their generalization power, resistance to the curse of dimensionality, and resistance to the attribute noise. In those experiments, we contrast the performance of the FSKM-k and FSKM-ϕ models against state-of-the-art models on twenty-six datasets. Furthermore, we present two novel positive definite kernels which belong to the class of kernels

[1]An initial version of this work was published in [5].

[2]Section A in the supplementar material unfolds those claims deeper

[3]One of the reasons for this is that non-singleton fuzzification operators from NSFS bring the unique perspective of modeling any crisp dataset as a dataset of vectors of fuzzy sets, bringing some modeling advantages that we discussed in the following paragraph.

TABLE II
STUDIED CONNECTIONS BETWEEN NON-SINGLETON MAMDANI FUZZY SYSTEMS AND KERNEL MACHINES

Model	Description
FSKM-k	Connection between a *non-normalized* fuzzy-basis-function expansion of non-singleton Mamdani fuzzy systems and kernel machines
FSKM-ϕ	Connection between a *normalized* fuzzy-basis-function expansion of non-singleton Mamdani fuzzy systems and kernel machines

on fuzzy sets [7], [8]: the *non-singleton kernel* and the *Fuzzy-Basis-Function* kernel, both introduced in Theorems 1 and 3.

A remark about non-singleton fuzzification operators[4]
Non-singleton fuzzification operators characterize NSFSs (see Section III-A, Definiton 3). In this work, we research the connection under the perspective of Non-singleton fuzzification operators instead of the widely used singleton fuzzification operators because the former brings the unique perspective of modeling any crisp data set as a dataset of vectors of fuzzy sets (a.k.a fuzzy dataset, see Section IV-A, Equation (18)). Consequently, any crisp input into the system is transformed into a vector of fuzzy sets by the non-singleton fuzzifier, and such vectors of fuzzy sets can be understood as logical premises matching the spirit of the generalized modus pones in fuzzy logic [9] (see Definition SM-14). Furthermore, because of the Representer Theorem of kernel methods, arbitrary kernel algorithms can select a subset of vectors of fuzzy sets from the fuzzy dataset — which we get by applying a non-singleton fuzzifier operator on the original training set pointwise — for modeling the *rule antecedents*, as we described in Section IV. Finally, optimal fuzzy dataset's parameters are data-driven estimated because of the kernel method pipeline and nested-cross validation procedures as we showed in Section V.

A remark about the word kernel. The fuzzy set community widely uses the word kernel to denote a crisp subset of a fuzzy set where all the elements have unity membership degrees. In machine learning, we use it to denote kernel functions used for statistical modeling. In this paper, we deal strictly with positive-definite kernels, consequently:

Kernel Machines $\overset{\text{def}}{=}$ regularized machine learning models with positive-definite kernels.

A. Paper organization

Section II introduces the main concepts of kernel machines, namely the kernel method, positive definite kernels, kernel algorithms, and the Representer theorem; Section III introduces fuzzy sets, non-singleton Mamdani fuzzy systems, and the Fuzzy-Basis-Function expansion of fuzzy systems; Section IV describes the Fuzzy-System Kernel machines based on the researched connections; Section A describes in detail the claimed benefits of studying this two-way connection;

[4]Singleton fuzzy systems and NSFS are equivalent under this particular setup: Gaussian membership functions with equal spread parameters, and product T-norm [1].

TABLE III
MATHEMATICAL NOTATIONS

Description	Notation	Definition
ith measured values	x_i (scalar), $\boldsymbol{x}_i \equiv (x_{i1}, \ldots, x_{iD})^T$	
input variable	X (scalar), $\boldsymbol{X} \equiv (X_1, \ldots, X_D)^T$	
ith output measured value	y_i	
output variable	Y	
input/output space	$\mathcal{X}$ and $\mathcal{Y}$	
fuzzy sets	Z, A, C	Sec. III
vector of fuzzy sets	$\mathbf{Z} \equiv (Z_1, \ldots, Z_D)^T$	
non-singleton fuzzy set	$Z(\cdot\|x_i) : \mathcal{X} \to [0, 1]$	Def. 3
vector of non-singleton fuzzy sets	$\mathbf{Z}_{\boldsymbol{x}} \equiv (Z(\cdot\|x_1), \ldots, Z(\cdot\|x_D))^T$	
T-norm operator	$T, \star$	Def. SM-13
function to be learned	$h : \mathcal{X} \to \mathcal{Y}$	
fuzzy-basis-function (FBF)	$\phi, \psi : \mathcal{X} \to \mathbb{R}$	Eq. 15
kernel function	$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$	Def. 1
kernel matrix	K	Sec. II-A
Reproducing Kernel Hilbert Space (RKHS)	$\mathcal{H}$	App. B

Section V presents a set of experiments and Section VI presents concluding remarks. Advanced concepts and proofs are provided in the supplementary material. Finally, Table III shows the mathematical notation used in this paper.

II. KERNEL MACHINES

Kernel machines are a class of models used for statistical machine learning. The kernel method, depicted in Figure 1, provides the pipeline for constructing kernel machines. Kernel functions are the core concept of this methodology because 1) they provide a way for estimating high dimensional features of the data being analyzed, and 2) they induce a high dimensional space where algorithms are defined for analyzing patterns in data. The next sections review positive definite kernels and their use in Machine learning through the kernel method.

A. Positive-definite Kernels

Positive-definite kernels play an important role in fields such as analysis, approximation theory, engineering, geostatistics, probability theory and statistics [2], [10]. Although kernel functions can be complex-valued or operator-valued, the most common kernels in machine learning are real-valued functions of the form: $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a non-empty set.

Definition 1 (Positive-definite kernels)**.** Let $\mathcal{X}$ be a non-empty set. Then, a symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive-definite kernel if it satisfies:

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j k(x_i, x_j) \geq 0, \tag{1}$$

for all $N \in \mathbb{Z}^+$, all $\alpha_i, \alpha_j \in \mathbb{R}$ and, all $x_i, x_j \in \mathcal{X}$.

An attractive property of a positive-definite kernel is that it is a *reproducing kernel* of a Reproducing Kernel Hilbert

Space (RKHS) $\mathcal{H}$ (See Supplemental Material B for the main definitions). The *kernel trick*[5]:

$$k(x, x') = \langle k(\cdot, x), k(\cdot, x') \rangle_{\mathcal{H}}, \tag{2}$$

is a practical result in machine learning for the characterization of positive-definite kernels as reproducing kernels of RKHSs — allowing an *implicit* inner product computation between the high dimensional representations $k(\cdot, x), k(\cdot, x') \in \mathcal{H}$ — a.k.a., feature vectors or representative functions — of data points $x, x' \in \mathcal{X}$. RKHSs are important for statistical Machine Learning because:1) if two functions are close in the sense of a norm then their function images are also close (i.e., a sequence converging in the norm also converges pointwise); 2) it is possible to implicitly perform inner products in $\mathcal{H}$ as $k(x, x') = \langle k(\cdot, x), k(\cdot, x') \rangle_{\mathcal{H}}$; and, 3) the evaluation of the estimated function h on a point x is interpreted as $h(x) = \langle h, k(\cdot, x) \rangle_{\mathcal{H}}$. This leverages: a) the design of geometrical algorithms (e.g., Support Vector Machines, Minimum Enclosing Balls); b) the extension of widely-known methods (Kernel PCA, Kernel Ridge Regression); and, c) the design of regularizers that explicitly penalize the function that is being approximated. Examples of positive-definite kernels $k(x, x')$ are: the linear kernel $x^T x'$, the polynomial kernel $(x^T x' + 1)^\gamma, \gamma > 0$, the Gaussian kernel $\exp(\gamma \|x - x'\|^2), \gamma > 0$, kernels on measures,on strings,on graphs, among others [3], [4].

B. The Kernel Method

The kernel method is a methodology for constructing kernel machines. Figure 1 presents the workflow and main concepts of this methodology. We review each of them as follows:

1) The data:

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \subset \mathcal{X} \times \mathcal{Y}, \tag{3}$$

The input space $\mathcal{X}$ has no topological restrictions. It can be any collection of elements representing real phenomena, e.g., a set of vectors on $\mathbb{R}^D$, graphs, probability distributions, etc.

2) The kernel function and the kernel matrix: The next step of the kernel method is to construct a similarity matrix K between elements of the dataset using a symmetric positive-definite kernel, i.e., $K_{ij} = k(x_i, x_j)$. The matrix K is symmetric and positive-semidefinite and is a core concept in the design of kernel machine algorithms, because it can be interpreted as *similarity matrix* between observations, i.e., $K_{ij} = k(x_i, x_j)$ informs how similar x_i is to x_j. Also, the kernel matrix plays an important role in the practical implementation of this models assuring the convexity of the objective function of kernel machine models given by (4).

3) Kernel algorithms: In Machine learning, the purpose of the kernel method is to create a model, the kernel machine, that estimates as best as possible a functional relationship $h : \mathcal{X} \to \mathcal{Y}$ from the dataset $\mathcal{D}$. Since many solutions are possible; the kernel method seeks solutions with high generalization (a.k.a., predictive power) [6]. Those solutions are estimated within the RKHS, which is the functional space induced by the kernel. To do that, kernel methods use a loss function and a regularizer to control the capacity of the learned function. Thus, kernel algorithms seek to find a solution $h^* : \mathcal{X} \to \mathcal{Y}$ by solving the following optimization problem:

$$\arg\min_{h \in \mathcal{H}} \left[\Omega(\lambda, \|h\|_{\mathcal{H}}) + \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(y_i, h(x_i)) \right] \tag{4}$$

where $\mathcal{H}$ is an RKHS, λ is a non-negative regularization parameter, Ω is a regularization function and, $\mathcal{L}$ is a loss function that measures the quality of h in terms of the error between the observed y_i and the predicted $h(x_i)$ value.

4) Solution via the Representer Theorem: The Representer Theorem [3] characterizes the solution of (4) in terms of the data sample $\mathcal{D}$, a symmetric positive-definite kernel k, an empirical loss L and a regularizer Ω. It states that any minimizer h^* of (4) has the form:

$$h^* = \sum_{i=1}^{N} \alpha_i k(\cdot, x_i) \tag{5}$$

where $\alpha_i \in \mathbb{R}$. From that, it follows that the kernel method produces kernel machines with regularizers explicitly penalizing the function h in terms of the kernel k. For instance, $\Omega(\lambda, \|h\|_{\mathcal{H}}) = \lambda \langle h, h \rangle_{\mathcal{H}}$ can be written in terms of kernel evaluations as:

$$\begin{aligned} \lambda \langle h, h \rangle_{\mathcal{H}} &= \lambda \langle \sum_{i=1}^{N} \alpha_i k(\cdot, x_i), \sum_{j=1}^{N} \alpha_j k(\cdot, x_j) \rangle_{\mathcal{H}} && (6) \\ &= \lambda \sum_{ij}^{N} \alpha_i \alpha_j \langle k(\cdot, x_i), k(\cdot, x_j) \rangle_{\mathcal{H}} && (7) \\ &= \lambda \sum_{ij}^{N} \alpha_i \alpha_j k(x_i, x_j) && (8) \end{aligned}$$

Note that kernel machines are modular, i.e., it is always possible to change the kernel function without changing the algorithm. Table SM-1 shows widely-used kernel machines. Useful references are [2]–[4].

III. Fuzzy Sets and Systems

Fuzzy systems approximate a function $h : \mathcal{X} \to \mathcal{Y}$ by applying operations from fuzzy set theory and fuzzy logic. Many fuzzy systems are universal approximators of functions [13]–[15], and have been used successfully in robotics, control, dynamic systems, machine learning, deep learning, intelligent systems [16]–[19], etc. At the heart of fuzzy systems is the concept of a (type-1) fuzzy set, originally defined by Zadeh as a mapping $Z : \mathcal{X} \to [0, 1]$, where $Z(x)$ is interpreted as the membership degree of element x in the fuzzy set Z, also called membership function, that plays an analogous role to the characteristic function in classic set theory [20]. Fuzzy sets are used in fuzzy logic to extend Boolean logic to a $[0, 1]$-valued logic, and, according to Dubois and Prade [21], they have three semantics: 1) similarity between a value and a prototype; 2) degree of uncertainty about an ill-known value; and 3) preference. Generalizations of fuzzy sets include L-fuzzy sets [22]; type-2 fuzzy sets, among others [23]. In the

[5] $k(\cdot, x)$ means that one fixes an $x \in \mathcal{X}$ of the second argument of $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, and varies the variable in the first argument.

[6] This differs from statistical descriptive modeling which aims at function estimation for description purposes, in which generalization is not important. See a related discussion in [12].

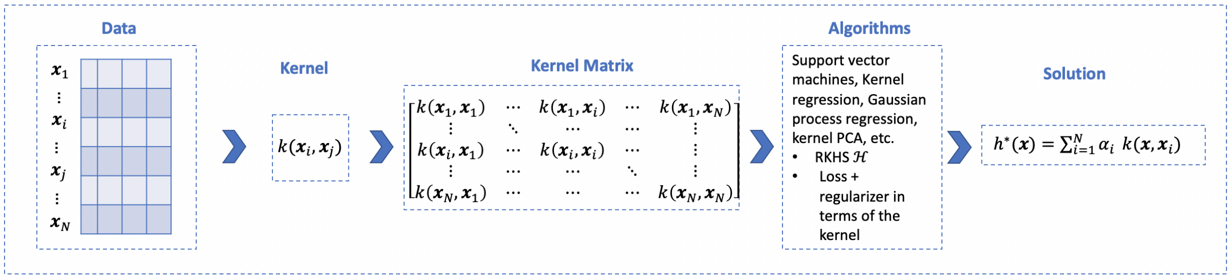
Data
x_1
x_i
x_j
x_N
Kernel
$k(x_i, x_j)$
Kernel Matrix
$\begin{bmatrix} k(x_1,x_1) & \cdots & k(x_1,x_i) & \cdots & k(x_1,x_N) \\ \vdots & \ddots & \cdots & \cdots & \vdots \\ k(x_i,x_1) & \cdots & k(x_i,x_i) & \cdots & \vdots \\ \vdots & \cdots & \cdots & \ddots & \vdots \\ k(x_N,x_1) & \cdots & \cdots & \cdots & k(x_N,x_N) \end{bmatrix}$
Algorithms
Support vector machines, Kernel regression, Gaussian process regression, kernel PCA, etc.
RKHS $\mathcal{H}$
Loss + regularizer in terms of the kernel
Solution
$h^*(x) = \sum_{i=1}^{N} \alpha_i \, k(x, x_i)$

Fig. 1. Workflow and related concepts of the kernel method. Figure adapted from [11]

remainder of the paper, we are going to limit the discussion to $\mathcal{X} = \mathbb{R}^D$ and $\mathcal{Y} = \mathbb{R}$; and to type-1 fuzzy sets.

A. *Non-singleton Mamdani Fuzzy Inference Systems (NSFS)*

A Mamdani Fuzzy inference system [24] is a rule-based machine learning model that uses data or/and expert knowledge for setting its rules, and several concepts from fuzzy logic and fuzzy set theory for defining the model's structure and performing inference[7]. This system has four main components: 1) a fuzzifier operator that transforms the real input values to input-fuzzy sets, because a fuzzy system uses fuzzy sets for all its internal operations; 2) a rule base composed of IF-THEN rules; 3) an inference engine that uses fuzzy logic principles to perform inference between an input-fuzzy set and an IF-THEN rule, resulting in a fired rule output-fuzzy set[8], and; 4) a defuzzification operator that maps all fired rule output-fuzzy sets into a real number.

Different choices for those components lead to different kinds of Mamdani fuzzy systems, e.g., in this work, we use Mamdani fuzzy systems that use a *non-singleton* fuzzifier operator resulting in *non-singleton Mamdani Inference systems* (NSFS) [6]. Also, we focus on a defuzzification operator that leads to a Fuzzy-Basis-Function (FBF) expansion that are universal approximators [1], [14], [15].

A core concept of NSFSs is the notion of IF-THEN rules:

Definition 2 (IF-THEN rule)**.** A generic IF-THEN rule for a Mamdani fuzzy system is $(l = 1, \dots, L)$:

$$\text{IF } X_1 \text{ is } A_{l1} \text{ and} \dots \text{and } X_D \text{ is } A_{lD} \text{ THEN } Y \text{ is } C_l \tag{9}$$

where X_d $(d = 1, 2, \dots, D)$ are the real input variables; Y is the real output variable; fuzzy sets $A_{ld} : \mathbb{R} \to [0, 1]$ belong to the first-order uncertainty partition[9] of variable X_d and; $C_l : \mathbb{R} \to [0, 1]$ are fuzzy sets belonging to the first-order uncertainty partition T_y of variable Y.

There are many methodologies for constructing those IF-THEN rules [1], ranging from using expert knowledge (See Example SM-1 in the Supplemental Material) to full data-driven approaches. Moreover, each IF-THEN rule models a *fuzzy implication*. There are several implication operators used in fuzzy logics, however Mamdani fuzzy systems use a T-norm operator for this (Definition SM-13 in Supplemental Material). This causes the loss of meaning of implication in logic but it adds practical value to using those systems in real world applications [1].

[7]Other types of fuzzy inference systems are discussed in [1].

[8]This is a *Generalized Modus Ponens* that uses an input-fuzzy set as *premise* and an IF-THEN rule as *implication* for activating a *consequent* which is a fuzzy set — see Def. SM-14.

[9]A first-order uncertainty partition is defined in Definition SM-11.

B. *Methodology for constructing Mamdani fuzzy systems based on Fuzzy-Basis-Function (FBF) expansion*

The methodology uses observed data and (optionally) expert knowledge to approximate a function $h : \mathbb{R}^D \to \mathbb{R}$. Figure 2 depicts the workflow and main concepts of this methodology. We review each of them as follows:

1) The data: Is the set of observed values:

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\} \subset \mathbb{R}^D \times \mathbb{R}, \tag{10}$$

2) Input expert knowledge: Optionally, expert knowledge can be provided as input to the system in the form of IF-THEN rules $(m = 1, \dots, M)$:

$$\text{IF } X_1 \text{ is } A_{m1} \text{ and} \dots \text{and } X_D \text{ is } A_{mD} \text{ THEN } Y \text{ is } C_m \tag{11}$$

3) Rules: A rule-base is constructed using the input data, resulting in a set of rules as in (9). Additionally, if expert IF-THEN rules are provided as inputs, those rules are added to the rule-base.

4) Fuzzy system choices: A fuzzy system has several hyperparameters, e.g., the number of fuzzy sets per dimension for modeling the fuzzy sets in the antecedent part of IF-THEN rules; the shape of fuzzy sets; the T-norm operator for modeling the word *and*; and the choice of a fuzzifier. In this work, we are interested in *non-singleton* fuzzifiers because of the reasons exposed in the introduction section.

Definition 3 (non-singleton fuzzifier)**.** Given an input value $\boldsymbol{x} \equiv (x_1, \dots, x_D) \in \mathbb{R}^D$ to the fuzzy system, a non-singleton fuzzifier maps each value $x_d \in \mathbb{R}$ $(d = 1, \dots, D)$ of $\boldsymbol{x}$ to the *non-singleton fuzzy set* $Z_d(\cdot|x_d) : \mathbb{R} \to [0, 1]$, which is a fuzzy set such that $Z_d(X_d|x_d) = 1$ when $X_d = x_d$ and $Z_d(X_d|x_d)$ decrease from unity as X_d moves away from x_d.

In the spirit of Fuzzy logic's Generalized Modus Ponens (Definition SM-14) the non-singleton fuzzifier's output: $\mathbf{Z}_{\boldsymbol{x}} \equiv (Z(\cdot|x_1), \dots, Z(\cdot|x_D))^T$ models the *premise*:

$$X_1 \text{ is } Z_1 \text{ and} \dots \text{and } X_D \text{ is } Z_D \tag{12}$$

which is used jointly with the rules in (9) to do inference (see Supplemental Material C-A). Thus, the interaction between the

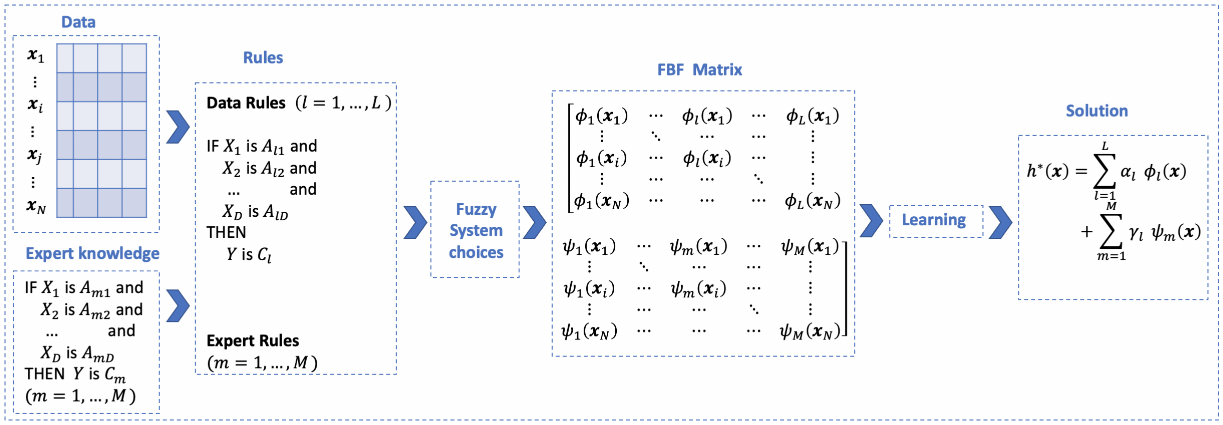
Data
x_1
x_i
x_j
x_N
Expert knowledge
IF X_1 is A_{m1} and X_2 is A_{m2} and … and X_D is A_{mD} THEN Y is C_m $(m = 1,\dots,M)$
Rules
Data Rules $(l = 1,\dots,L)$
IF X_1 is A_{l1} and X_2 is A_{l2} and … and X_D is A_{lD} THEN Y is C_l
Expert Rules $(m = 1,\dots,M)$
Fuzzy System choices
FBF Matrix
$\phi_1(x_1)$ … $\phi_l(x_1)$ … $\phi_L(x_1)$
$\phi_1(x_i)$ … $\phi_l(x_i)$
$\phi_1(x_N)$ … $\phi_L(x_N)$
$\psi_1(x_1)$ … $\psi_m(x_1)$ … $\psi_M(x_1)$
$\psi_1(x_i)$ … $\psi_m(x_i)$
$\psi_1(x_N)$ … $\psi_M(x_N)$
Learning
Solution
$h^*(x) = \sum_{l=1}^{L} \alpha_l \phi_l(x) + \sum_{m=1}^{M} \gamma_l \psi_m(x)$

Fig. 2. Function approximation via Fuzzy-Basis-Function expansion of Mamdani fuzzy systems

premise and a rule's antecedent (Supplemental Material C-B) is characterized by:

$$\begin{aligned} &sup_{\boldsymbol{X}\in\mathbb{R}^D} T_{d=1}^{D} Z_d(X_d|x_d) \star T_{d=1}^{D} A_{ld}(X_d) \\ &\equiv T_{d=1}^{D} \sup_{X_d\in\mathbb{R}} Z_d(X_d|x_d) \star A_{ld}(X_d) \end{aligned} \tag{13}$$

where T and $\star$ denote the t-norm operator (Definition SM-13) and sup denotes the suppremum operation.

Another choice is the type of defuzzification operator [1], [25]. In this work we chose the Center-of-Sets defuzzifier [1] because it *leads* to the FBF expansion of fuzzy systems.

5) Fuzzy-Basis-Function matrix: An important result in fuzzy systems is that Mamdani fuzzy systems are universal approximators of functions that can be expressed as the following FBF expansion [1], [14], [15]:

$$h(\boldsymbol{x}) = \sum_{l=1}^{L} \alpha_l \phi_l(\boldsymbol{x}) \tag{14}$$

where ϕ_l functions are called fuzzy-basis-functions[10], $\alpha_l \in \mathbb{R}$, and:

$$\phi_l(\boldsymbol{x}) = \frac{T_{d=1}^{D} \sup_{X_d\in\mathbb{R}} Z_d(X_d|x_d) \star A_{ld}(X_d)}{\sum_{l=1}^{L} T_{d=1}^{D} \sup_{X_d\in\mathbb{R}} Z_d(X_d|x_d) \star A_{ld}(X_d)} \tag{15}$$

In (14) and (15), index $l = 1,\dots,L$ is related to the IF-THEN rules. Thus, the FBF matrix is defined as $\Phi_{il} = \phi_l(\boldsymbol{x}_i)$. Moreover, if an additional set of $m = 1,\dots M$ expert or knowledge-driven IF-THEN rules are provided then (14) can be rewritten as:

$$h(\boldsymbol{x}) = \sum_{l=1}^{L} \alpha_l \phi_l(\boldsymbol{x}) + \sum_{m=1}^{M} \gamma_m \psi_m(\boldsymbol{x}), \tag{16}$$

where coefficients $\gamma_m \in \mathbb{R}$ and FBF ψ_m are related to the provided expert IF-THEN rules. The FBF matrix is $[\Phi\ \Psi]$, where $\Psi_{im} = \psi_m(\boldsymbol{x}_i)$.

[10]It also can be understood as a TSK fuzzy system with constant α_l. However, we avoided calling TSK in our work because a Mamdani has only one FBF per rule — as is the case of the paper, whereas TSK fuzzy systems have $d+1$ FBFs per rule, leading to an FBF expansion with $L(d+1)$ terms—we let such analysis as future work.

6) Learning: Fuzzy systems use data for optimizing a loss function regarding a task (regression, classification, forecasting, etc.) with respect to the fuzzy system's parameters, i.e., fuzzy set parameters and rule parameters. Moreover, regularization can be used by including a term in the loss function to account for fuzzy system complexity.

7) Solution: The solution given by (15) (or (16) if expert IF-THEN rules are used) is a fuzzy basis function expansion. Even though one begins with rules and fuzzy sets, one can write formulas for this FBF expansion.

IV. Fuzzy-System Kernel Machines

This section introduces the Fuzzy-System kernel machines (FSKM) produced from investigating two connections between kernel machines and non-singleton Mamdani fuzzy systems (NSFS). The idea is to relate the fuzzy-basis-function expansion in (15), positive definite kernels, and the Representer Theorem of kernel methods. Table II shows the two explored connections and the resulting FSKM-k and FSKM-ϕ models, introduced in Sections IV-B and IV-C respectively. We begin by showing how to adapt the kernel method for working with fuzzy sets and systems concepts.

A. The kernel method with kernel on fuzzy sets

The kernel method works on any kind of input datasets (sets, measures, graphs, etc.). Figure 3 depicts how the kernel method can work with concepts from fuzzy sets and systems. Firstly, given a crisp dataset containing observations:

$$\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \mid i = 1,2,\dots,N\} \subset \mathbb{R}^D \times \mathbb{R} \tag{17}$$

and a non-singleton fuzzifier, we construct the following *fuzzy dataset*:

$$\mathcal{D}_{\mathcal{F}} = \left\{ \left(\mathbf{Z}_{\boldsymbol{x}_i}, y_i\right) \mid i = 1,2,\dots,N \right\} \subset \mathcal{F}(\mathbb{R})^D \times \mathbb{R}, \tag{18}$$

where

$$\mathbf{Z}_{\boldsymbol{x}_i} \equiv \left(Z_{i1}(\cdot|x_{i1}),\dots,Z_{iD}(\cdot|x_{iD})\right)^T \in \mathcal{F}(\mathbb{R})^D$$

is a D dimensional vector of non-singleton fuzzy sets for the observed data point $\boldsymbol{x}_i \equiv (x_{i1},\dots x_{iD})^T$. Notation $\mathcal{F}(\mathbb{R})$ is

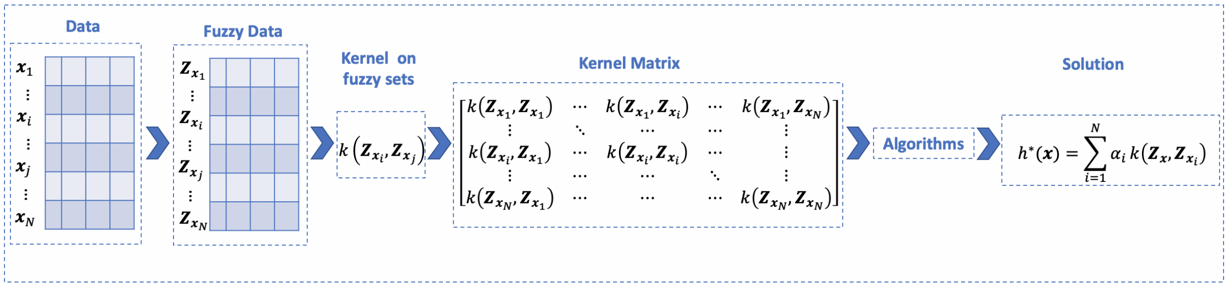
Data
Fuzzy Data
Kernel on fuzzy sets
Kernel Matrix
Algorithms
Solution
$h^*(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i k(\boldsymbol{Z}_{\boldsymbol{x}}, \boldsymbol{Z}_{\boldsymbol{x}_i})$

Fig. 3. The kernel method with kernel on fuzzy sets

the set of all the fuzzy sets on $\mathbb{R}$. Note that (18) gives a fuzzified version of the original data set in (17). In this version, each value per dimension from (17) is transformed into a (non-singleton) fuzzy set around that value (Definition 3).

To construct an $N \times N$ kernel matrix using the fuzzy dataset we need a kernel defined on the set of fuzzy sets, i.e., a kernel on fuzzy sets. Therefore, depending on the task (classification, regression, forecasting, interpolation/kriging, data description, dimensionality reduction, etc.), a kernel algorithm is chosen (SVM, kriging, SVDD, kernel PCA, etc.). Finally, because of the representer theorem, the solution is stated as a function of the training data, in this case, the fuzzy dataset:

$$h^*(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i k(\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}_i}) \tag{19}$$

Based on this, the next sections describe two connections based on two novel kernels on fuzzy sets.

B. The connection between a non-normalized FBF expansion of non-singleton Mamdani fuzzy systems and kernel machines

This section explores the first connection, leading to the formulation of the FSKM-k model: In Theorem 1, we postulate a new kernel on fuzzy sets based on a non-normalized FBF of non-singleton Mamdani fuzzy systems (Equation (15) without the denominator), we define FSKM-k models in Definition 4 and postulate the connection and functional equivalence in Theorem 2. Also, we describe a top-down approach for designing fuzzy systems under this perspective.

Theorem 1 (Non-singleton kernel on fuzzy sets)**.** Let $\mathcal{F}(\mathbb{R})$ denote the sets of fuzzy sets on $\mathbb{R}$ and $\mathcal{F}(\mathbb{R})^D$ its cartesian product, where D is the data dimension. The mapping $k : \mathcal{F}(\mathbb{R})^D \times \mathcal{F}(\mathbb{R})^D \to [0, 1]$ defined by

$$k(\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}'}) = \sup_{\boldsymbol{X} \in \mathbb{R}^D} T_{d=1}^{D} Z_d(X_d|x_d) \star T_{d=1}^{D} Z'_d(X_d|x'_d), \tag{20}$$

where

$$\mathbf{Z}_{\boldsymbol{x}} \equiv \big(Z_1(\cdot|x_1), \ldots, Z_D(\cdot|x_D)\big)^T \tag{21}$$

and,

$$\mathbf{Z}_{\boldsymbol{x}'} \equiv \big(Z'_1(\cdot|x_1), \ldots, Z'_D(\cdot|x_D)\big)^T \tag{22}$$

is a positive definite kernel for any T-norm dimensional aggregation operator T, Gödel or product T-norm for T-norm operator $\star$, and normal and convex fuzzy sets with infinite support for symmetric non-singleton fuzzy sets Z_d and Z'_d, $(d = 1, \ldots, D)$[11].

Thus, we have the following machine learning model:

Definition 4 (FSKM-k model)**.** A FSKM-k model is a regularized kernel machine trained with the non-singleton kernel on fuzzy sets.

Thus, given a loss function, a regularizer, a crisp dataset as in (17), a fuzzy data as in (18) and a non-singleton kernel on fuzzy sets (20), by the Representer theorem for kernel machines, the solution of a regularized kernel machine with those components is given by (19). Observe that the domain of the approximated function by the kernel method h is $\mathbb{R}^D$. Moreover, the right hand of (19) shows that a non-singleton fuzzifier acts on the input and elements of the training set, i.e., $\boldsymbol{x}$ and $\boldsymbol{x}_i$, transforming them into fuzzy sets, which are arguments of the non-singleton kernel on fuzzy sets. In practice, this leads to a machine learning pipeline where non-singleton fuzzification is a data pre-processing step.[12]

Top-down approach for designing fuzzy systems: Fuzzy systems are *rule-based learning models*: The learning process is based on optimizing the parameters of a set of IF-THEN rules to match the functional dependence between input and outputs and, the inference process uses those optimized rules for generalizing to unseen inputs. Those rules are constructed from available data and expert knowledge, and are prototypes capturing the patterns in data. On the other hand, because of the Representer Theorem, kernel machines are *instance-based learning models*, in the sense that they need to memorize some instances or observations (a subset of the available data) and use those memorized values for inference.

Due to the rule-based vs. instance-base modeling difference, we make the following assumption to make possible the first connection between kernel machines with non-singleton kernel and, non-singleton fuzzy systems:

Assumption 1. Each memorized observation by a FSKM-k model can be interpreted as an antecedent part of an IF-THEN rule.

[11]We can eliminate the infinite support requirement by requiring that each $Z_d(\cdot|x_d)$, $(d = 1, \ldots, D)$ overlaps with their nearest neighbors within their corresponding first-order uncertainty partition (Definition SM-11) to assure that the supremum exists.

[12]Feature extraction, dimensionality reduction, feature union, etc., are other examples of ML pre-processing steps.

Notice that the set of the memorized instances or observations by the solution h^* from (19) is the set:

$$\mathcal{M} = \left\{ \mathbf{Z}_{\boldsymbol{x}_i} \mid \alpha_i \neq 0, i = 1, \ldots, N \right\} \tag{23}$$

This interpretation provides a *top-down* approach to designing fuzzy systems instead of a local approach [26], i.e., the kernel machine memorizes some observations of data and sets those observations to be fuzzy-rule antecedents (top-down), instead of deriving the antecedent fuzzy sets and rules by analyzing the coverage of the rules with respect to the each dimension of the data (local-design).

Assumption 1 has two consequences: 1) the resulting kernel machine is a rule-based model that can be viewed as an NSFS, consequently a set of IF-THEN rules can be retrieved from those models; and, 2) we can learn the antecedent part of IF-THEN rules of a fuzzy system by using the memorized observations from the kernel machine, consequently, the resulting NSFS is a regularized kernel machine estimating its solution in an RKHS. That is summarized in the following result.

Theorem 2 (Connection and functional equivalence)**.** If we set the number of rules L of a NSFS to be the cardinality of the memorized instances by the solution of a kernel machine given by (19), i.e. $L = card(\mathcal{M})$ and, if we let each element from the set of memorized instances define the antecedent fuzzy sets of those rules. i.e., $(A_{l1}, \ldots, A_{lD}) \in \mathcal{M}$ Then:

- The underlying kernel machine is a NSFS
- The resulting NSFS is a kernel machine

Consequently, the set of memorized instances $\mathcal{M}$ in (23) defines the following rule base $(l = 1, \ldots, L)$:

$$\text{IF } X_1 \text{ is } A_{l1} \text{ and} \ldots \text{and } X_D \text{ is } A_{lD} \text{ THEN } Y \text{ is } \alpha_l, \tag{24}$$

where each $(A_{l1}, \ldots, A_{lD})$ is an element of $\mathcal{M}$ and α_l is its respective non-zero coefficient value. Consequently, those rules can be retrieved from a trained kernel machine with a non-singleton kernel on fuzzy sets or can be used for defining the following non-singleton fuzzy system with non-normalized FBF expansion:

$$h(x) = \sum_{l=1}^{L} \alpha_l \big(T_{d=1}^{D} \sup_{X_d \in \mathbb{R}} Z_d(X_d|x_d) \star A_{ld}(X_d) \big) \tag{25}$$

The learning process estimates the optimal α_l values for a given loss function, regularizer, kernel, and fuzzification parameters. Observe that this learning formulation generalizes the *support vector learning* approach [27]–[30][13] because it is not limited for learning singleton fuzzy systems using support vector machines. On the contrary, we can train an NSFS with any regularized kernel machine (See Table SM-1). Also, (25) explicitly modeled the premise in (12) with non-singleton fuzzy sets instead of singleton fuzzy sets.

C. The connection between a normalized FBF expansion of non-singleton Mamdani fuzzy systems and kernel machines

There is no direct connection between kernel machines and the normalized FBF-expansion of a NSFS ((14) and (15)) because the resulting FBF matrix $\Phi_{il} = \phi_l(\boldsymbol{x}_i)$ is not symmetric, which is a requirement for defining reproducing kernels. Here, we formulate a symmetric kernel that uses the FBF formulation. Kernel machines using those kernels can be understood as the sum of two NSFSs.

Theorem 3 (kernel k_ϕ on fuzzy sets)**.** Let $\mathcal{D}_{\mathcal{F}}$ be fuzzy data defined by (18), such that $\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j}$ $(i, j = 1, \ldots, N)$ are in it. Let k be the non-singleton kernel on fuzzy sets given by (20). The mapping $k_\phi : \mathcal{F}(\mathbb{R})^D \times \mathcal{F}(\mathbb{R})^D \to [0, 1]$ defined by

$$k_\phi(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j}) = \frac{k(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j})}{\sum_{j=1}^{N} k(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j})} + \frac{k(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j})}{\sum_{i=1}^{N} k(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j})}, \tag{26}$$

is a positive definite kernel.

Thus, we have the following machine learning model:

Definition 5 (FSKM-ϕ model)**.** A FSKM-ϕ model is a regularized kernel machine trained with kernel k_ϕ.

Observe that the kernel in (26) is the sum of two basis functions, as in (15), but is written explicitly as a function of vectors of fuzzy sets instead of crisp values: $k_\phi(\mathbf{Z}_{\boldsymbol{x}_i}, \mathbf{Z}_{\boldsymbol{x}_j}) \equiv \phi_j(\boldsymbol{x}_i) + \phi_i(\boldsymbol{x}_j)$. Also, by the Representer theorem for kernel machines, a FSKM-ϕ model has a similar solution as the one in (19), i.e., $h^*(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i k_\phi(\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}_i})$. Moreover, the set of the memorized instances by that kernel machine is also defined by the set $\mathcal{M}$ in (23).

Connection and Functional equivalence: A kernel machine trained with kernel k_ϕ can be interpreted as the sum of two NSFS with FBF-expansion because[14]:

$$\begin{aligned} h^*(\boldsymbol{x}) &= \sum_{i=1}^{N} \alpha_i k_\phi(\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}_i}) && (27)\\ &\equiv \sum_{l=1}^{L} \alpha_l \phi_l(\boldsymbol{x}) + \sum_{l=1}^{L} \alpha_l \phi_{\boldsymbol{x}}(\boldsymbol{x}_l) && (28)\\ &\equiv h_1^*(\boldsymbol{x}) + h_2^*(\boldsymbol{x}) && (29) \end{aligned}$$

Applying similar arguments to those from Assumption 1 and Theorem 2 to the NSFS h_1 in (29) we have the following: The number of rules of the NSFSs h_1 is set equal to the cardinality of the memorized instances by the solution of a kernel machine given by (27), i.e. $L = card(\mathcal{M})$ and, each element from the set of memorized instances defines the antecedent fuzzy sets of those rules. Consequently, the rules for h_1 are defined as in (24) and, the non-singleton fuzzy sets in the input are interpreted as (12). However, the FBF ϕ for h_1 are given by (15) (normalized FBF).

On the other hand, the NSFS h_2 in (29) considers the memorized instances by the kernel machine (the rule antecedents in h_1) as inputs, i.e., each element of the set of memorized instances $\mathcal{M}$ define the premise: "X_1 is A_{l1} and…and X_D is A_{lD} ". Moreover, h_2 considers the test point $\boldsymbol{x}$ as a rule antecedent, i.e., the non-singleton fuzzy sets in $\mathbf{Z}_{\boldsymbol{x}}$ defines the rule

[13]See the discussion in Section IV-D.

[14]The index $(i = 1, \ldots, N)$ is replaced by $(l = 1, \ldots, L)$ because In (29) we use (26) and because Assumption 1.

IF X_1 is Z_1 and…and X_D is Z_D THEN Y is α_l, The study of the implications of this combination between h_1 and h_2 is left as future research.

In summary:

- a FSKM-ϕ model is the sum of two NSFSs
- The rules of the first NSFS, i.e., h_1 are estimated from the set of memorized instances by the FSKM-ϕ model.
- The second NSFS h_2 considers the test points as the rule antecedents and the memorized instances from the kernel machine as the inputs.

D. Related Work

Literature shows several works using fuzzy theory and positive kernel functions for doing machine learning [31]–[33]. As a result, some positive definite kernels can be viewed as fuzzy equivalence relations [34], [35] and a novel class of kernels defined on fuzzy sets was introduced in [7], [8], [36]. An interesting path of research of this synergy is studying the connection between the kernel matrix and the fuzzy system's IF-THEN rules, as we do in this work. The main motivation for this research direction was the training of fuzzy systems using the *support vector learning* approach, i.e., once a kernel machine is trained, the method uses all the *support vectors* (all the observations $\boldsymbol{x}_i$ with $\alpha_i > 0$ from (5)) to define the IF-THEN rules[15]. For instance, authors in [27] found that singleton fuzzy systems using fuzzy sets in the antecedent of IF-THEN rules, satisfying $A(x) = A(-x)$ and $A(0) = 1$, with A being a positive-definite function, define the following positive-definite kernel $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \prod_d^D A_d(x_{id} - x_{jd})$. After training a SVM with those kernels, the support vectors were used to identify the rule number and the parameters for the fuzzy sets in the antecedent part of IF-THEN rules. The same idea was used in [28] with A_d being Gaussian functions, also in [29] jointly with rule and feature ranking algorithms, and also in [30] but with kernel $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \prod_d^D A_d(x_{id}) \cdot A_d(x_{jd})$ under the interpretation of a fuzzy system as a neural network. Interestingly, this last work was that the connection between kernels and rule antecedents was used after clustering for rule definition and before rule reduction. It is worth mentioning that all those works use an FBF expansion of singleton fuzzy systems where the denominator of the resulting FBF expansion is always a positive number by construction, and it can always be ignored in classification tasks[16].

Our work generalizes these previous works in two ways: 1) we use non-singleton fuzzifiers (Definition 3) which are more general than the singleton fuzzifiers used in previous works[17] A consequence of this is that we can control the fuzzification level of the inputs, and we can formulate the connection in terms of the kernel on fuzzy sets; 2) Previous works only used SVM for support vector learning of fuzzy systems. We argue that any kernel machine (kriging, Gaussian process, SVD, kernel PCA, etc.) can be used as well for learning fuzzy systems.

Some Interesting works connect the kernel matrix with the consequent part of fuzzy systems, for instance [37] uses a linear kernel for learning the consequent parameter of a Takagi-Sugeno-Kang fuzzy system and [38] proposes to use a kernel with elements $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi_l(\boldsymbol{x}_i)\phi_l(\boldsymbol{x}_j)\boldsymbol{x}_i^T\boldsymbol{x}_j$ and SVM for learning the consequent parameters of IF-THEN rules. Other works in this direction are [39], [40].

V. Experiments

We conducted three experiments in supervised classification to quantify the performance of FSKM-k and FSKM-ϕ models in terms of 1) generalization power, 2) resistance to the curse of dimensionality, and 3) resistance to attribute-noise. We used a support vector machine algorithm as a kernel learning algorithm for training the FSKM-k, and FSKM-ϕ models[18]. For all the classifiers in the three experiments, we used a nested cross-validation procedure to get consistent statistical estimates which is more computationally expensive than single cross-validation methods, but it leads to less biased results [41]. Nested cross-validation is implemented via two cross-validation loops. The outer loop is in charge of computing the cross-validation score, and the inner loop is in charge of model selection. In this sense, we implemented model selection via a randomized grid search over the hyperparameters of all the classifiers. We also set as an additional hyper-parameter whether or not to scale all the data values into the $[0, 1]$-interval — for all the classifiers in the three experiments. We used five folds for both the inner and outer partitions of the nested cross-validation procedure for all the experiments. We also estimated the nested cross-validation score in terms of an accuracy metric[19]. Furthermore, we performed a statistical comparison of FSKM classifiers vs. competing methods on twenty-six classification datasets using the Hierarchical Bayesian test [42] which is recommended for performing classifier comparison through datasets when the results come from cross-validation procedures. Additionally, we report Bayesian signed-rank test results in the Supplemental Material.

A. Kernels and FSKM classifiers setup

We used a non-singleton kernel on fuzzy sets based on Gaussian membership functions and the product T-norm operator for all the experiments given by the following expression[20,21]:

$$k\big(\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}'}\big) = \exp\Big(-0.5\gamma \sum_{d=1}^{D} \frac{(x_d - x'_d)^2}{\sigma_d^2 + {\sigma'_d}^2}\Big), \tag{30}$$

where $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^D$, $\mathbf{Z}_{\boldsymbol{x}} \equiv (Z(\cdot|x_1), \ldots, Z(\cdot|x_D))^T$ and $\mathbf{Z}_{\boldsymbol{x}'}$ is defined similarly. Here, each fuzzy set $Z_d(\cdot|x_d)$ has

[15]To the best of our knowledge, previous works only use SVM for classification or regression.

[16]Because in classification only the sign of the output matters.

[17]A singleton fuzzifier transform the input x_d in the *singleton* fuzzy set $Z(X_d|x_d) = 1$ when $X_d = x_d$ and 0 otherwise. Note that a singleton fuzzifier is a particular case of a non-singleton fuzzifier.

[18]Depending on the task, note that it is possible to use other kernel algorithms, e.g., kernel ridge regression, Gaussian process regression, etc.

[19]We used the *scikit-learn* package for that purpose.

[20]The derivation of this expression appears in [1], [6] in the context of computing the firing level for rules in non-singleton fuzzy systems.

[21]We let the role of the use of membership functions other than Gaussians and with different spreads as future research.

TABLE IV
HYPER-PARAMETERS OF FSKM-k AND FSKM-ϕ MODELS

Model	Description
Fuzzification level	$\sigma_{id} \in \mathbb{R}^+$ (one for each scalar value x_{id})
Kernel parameter	$\lambda \in \mathbb{R}^+$
Regularization parameter	$\gamma \in \mathbb{R}^+$

Gaussian membership function with parameters x_d, σ_d, i.e., $Z_d(\cdot|x_d) = \exp(-0.5(\cdot - x_d)/\sigma_d^2)$. Additionally, we consider the kernel parameter $\gamma > 0$ to control the extent to which the two elements $\mathbf{Z}_{\boldsymbol{x}}, \mathbf{Z}_{\boldsymbol{x}'}$ are similar. Thus, we trained the FSKM-k with kernel (30), and the FSKM-ϕ with the kernel (26) using (30) as the base kernel. Table IV shows the FSKM's hyper-parameters that were optimized by random grid search in the model selection step of the nested cross-validation. We also consider the same fuzzification level for all the values[22].

B. Assessing the generalization power of FSKM classifiers

This experiment assesses the generalization power of the proposed FSKM models — in terms of the accuracy metric — against popular machine learning models. Table V contains classifiers details and, Table SM-2 shows the twenty-six classification datasets used in this experiment. We describe the data preprocessing, experimental design, and classifier results in the following paragraphs.

TABLE V
CLASSIFIERS USED TO QUANTIFY THE GENERALIZATION POWER OF FSKM CLASSIFIERS TRAINED WITH KERNEL MACHINES

Label	Classifier
FSKM-k	Sec IV-B
FSKM-ϕ	Sec IV-C
RF	Random Forest
BagC4.5	Bagging C4.5
MLP	Multilayer perceptron neural network
SVM_RBF	Support vector machine with RBF kernel
C4.5	C4.5 decision tree algorithm
NB	Naive Bayes
KNN	k-nearest neighbors

1) SMOTE data preprocessing: Some of the datasets in Table SM-2 are imbalanced, thus, as the goal of this experiment is to quantify the generalization power of the FSKM models and not to solve or deal with imbalanced problems, we proceed to transform the original datasets into balanced versions by applying the SMOTE algorithm [43] which produces balanced datasets versions from the original ones by oversampling the minority classes. That allows us to use the balanced version of the datasets for training the models and reporting the results with the accuracy metric. Table SM-2 reflects the data statistics for the datasets processed with the SMOTE algorithm.

2) Classifier comparison based on Bayesian analysis: Figure 4 shows the results for this experiment. The y-axis represents the models, and the y-axis represents the accuracy metrics in terms of *violin plots*, which describe the accuracies distribution across all the datasets estimated via a kernel density estimator. Figure SM-3 from the Supplemental Material shows the distribution of accuracies per individual dataset. Note that the five outer loops of the nested cross-validation procedure provide five cross-fold accuracies per model and dataset for a total of 130 accuracies for estimating each violin plot $(5(\text{folds}) * 26(\text{datasets}) = 130(\text{accuracies}))$.

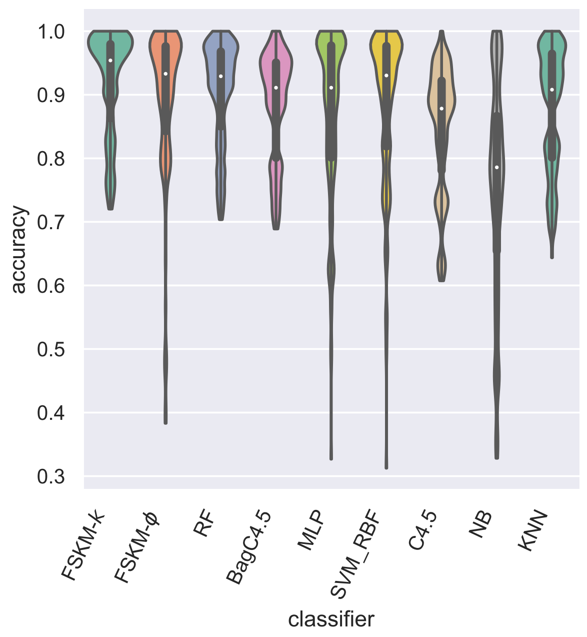

Fig. 4. Results for the generalization-power experiment. Each violin plot represents the distribution of accuracies across the twenty-six datasets.

We performed a statistical comparison between classifiers across all the datasets using the Hierarchical Bayesian test. This test uses the nested-cross validation accuracy values per classifier. Given two classifiers A and B, the output of that test are three probability values $\{P(A \gg B), P(A \ll B),$ and $P(A = B)\}$. Thus, given an arbitrary probability value ρ acting as a threshold, we can decide: $A \gg B$ if $P(A \gg B) > \rho$; $A \ll B$ if $P(A \ll B) > \rho$ or $A = B$ if $P(A = B)$. Additionally, the Hierarchical Bayesian test uses the *rope* parameter, which defines a region of practical equivalence for defining $P(A \ll B)$. We used $0, 0.01$ and 0.05 for the *rope* parameter. In the first case (rope = 0), there is no region of practical equivalence, which means that a classifier is always better than another. In the second and third cases (rope = 0.01

[22]Observe that a NSFS with Gaussian memberships functions sharing the same σ parameter is equivalent to a singleton fuzzy systems [1], consequently, by reparameterization kernel in (30) can be written equivalently as $\exp(-0.5\gamma' \sum_{d=1}^{D}(x_d - x_d')^2)$ with $\gamma' = \gamma * \frac{1}{2\sigma^2}$. This specific configuration leads to three theoretically equivalent classifiers[23]: the FSKM-k model, a singleton fuzzy system with RBF kernel (PDFC) model [27] and a support vector machine with RBF kernel. However, the FSKM-k model always will try to jointly optimize the fuzzification σ parameter in addition to the kernel bandwidth parameter γ from data, i.e., the σ fuzzification parameter constraint the possible choices for γ. Besides the advantages mentioned in the introduction, practical implications on this are that FSKM-k classifiers will be more resistant to attribute noise than other classifiers due to the non-singleton fuzzification step, as we observed in the experiments in Section V-D. We let as future research the role of different fuzzification parameters, membership functions, and ML tasks for FSMK models.

or rope = 0.05), two classifiers are equivalent if the difference of their accuracies is less than one or five percent, respectively.

Table VI shows the results estimated by the Bayesian Hierarchical test[24] By settign a threshold of $\rho = 0.95$, a value of T means that the base classifier (FSKM-k or FSKM-ϕ) is better because $P(\text{FSKM} \gg \text{other}) > 0.95$. A value of E means that the classifiers are equivalent $P(\text{FSKM} = \text{other}) > 0.95$ and, A value of F means that $P(\text{FSKM} \ll \text{other}) > 0.95$. Finally, symbol ? denotes that no decision can be made based on the threshold $\rho = 0.95$ because $P(\text{FSKM} \gg \text{other}), P(\text{FSKM} = \text{other}), P(\text{FSKM} \ll \text{other})$ are always less than $\rho = 0.95$.

TABLE VI
STATISTICAL COMPARISON OF CLASSIFIERS USING HIERARCHICAL BAYESIAN TESTS

Rope	0.00	0.01	0.05	0.00	0.01	0.05
Classifier	FSKM-ϕ			FSKM-k		
FSKM-k	?	E	E	-	-	-
FSKM-ϕ	-	-	-	?	E	E
RF	T	?	E	T	?	E
BagC4.5	T	T	E	T	T	E
MLP	T	E	E	T	?	E
SVM_RBF	?	E	E	?	E	E
C4.5	T	T	?	T	T	T
NB	T	T	T	T	T	T
KNN	T	?	E	T	T	E

3) Results and discussion: Figure 4 shows that — if we ignore the outliers — the distribution of accuracies is somewhat similar among the classifiers except for C4.5 and NB. Such information is well captured by the Hierarchical test in Table VI in the case of the FSKM-k classifier at rope 0.05. The Bayesian Hierarchical test shows us that at rope = 0 and with probability 0.95, both the FSKM-k and FSKM-ϕ classifiers *outperform* all the competing methods, with no-decision against SVM_RBF. Furthermore, at rope = 0.01, classifiers FSKM-k, FSKM-ϕ, and SVM_RBF are equivalent with probability 0.95, the beforementioned threshold of the test. Also, at this rope level, FSKM-ϕ is equivalent to MLP. This information is also visually suggested in Figure 4. We can conclude from the Bayesian Hierarchical test results that the proposed FSKM-k and FSKM-ϕ classifiers are competing methods of classical ML approaches on generic classification tasks. Also, there is no case where the proposed classifiers perform worse than competing methods. Finally, the Hierarchical Bayesian test gives a probability of $P(\text{FSKM-}k = \text{FSKM-}\phi) = 1.0$ at rope 0.01, which means that both classifiers have equivalent behavior. Based on this analysis, we conclude that both FSKM classifiers, FSKM-k and FSKM-ϕ have equivalent behavior.

C. Resistance to the curse of dimensionality

This experiment aims to empirically show that under the interpretation of FSKM as a fuzzy system, those fuzzy systems are resistant to the curse of dimensionality, i.e., a fuzzy system trained with kernel machines using the kernels proposed in this work inherits such property from kernel machines.

[24]Table SM-3 shows the rthe Bayesian signed-rank test results.

1) dataset: We generated binary-classification datasets starting from dimension 2 to dimension 100, which resulted in a total of 98 datasets. We kept the number of features per dimension to be the same number of informative features. This procedure was suggested by [44], and implemented in scikit-learn [45] by the procedure `make_classification`.

2) Experimental design and models: We used nested cross-validation as described in Section V-B. Table VII shows the fuzzy classifiers from the *fylearn* package and some classical Machine learning classifiers used in this experiment.

TABLE VII
CLASSIFIERES USED IN THE RESISTANCE OF THE CURSE OF DIMENSIONALITY EXPERIMENT

Label	Classifier	Reference
FSKM-k	NSFS + non-singleton kernel on fuzzy sets	this work
FSKM-ϕ	NSFS + kernel on fuzzy sets	this work
FPC	Fuzzy Pattern Classifier	[46]
MEC	Multimodal Evolutionary Classifier	[47]
FPT	Fuzzy Pattern Tree Top-Down Classifier	[48]
FRR	Fuzzy Reduction Rule Classifier	[49]
FPG	Fuzzy Pattern Classifier Genetic Alg.	[46]
FPTC	Fuzzy Pattern Tree Classifier	[50]
SVM_RBF	Support vector machine with RBF kernel	-
MLP	multilayer perceptron neural network	-
C4.5	C4.5 decision tree algorithm	-
BagC4.5	Bagging C4.5	-
NB	Naive Bayes	-
SGD	Linear models with stochastic gradient descent	-
RF	Random forest	-

3) Results and discussion: Each plot in Figure 5 shows the accuracy (y-axis) as a function of the data dimension (x-axis). Each line represents the accuracy values through the dimension and, its respective shadow region is the spread due to the distribution of accuracies from the five-folds of the outer-loop of the nested cross-validation procedure. We observe that the proposed FSKM classifiers are more resistant to the curse of dimensionality than the fuzzy classifiers: FPG, FPC, MEC, FPT (left plot) FRR, and FPTC (middle plot). Observe in the middle plot of Figure 5 that the FSKM-k classifier is more resistant than C4.5, NB, and BagC4.5. Finally, the right plot of Figure 5 shows that FSKM-k is better than MLP, RF, and SGD in higher dimensions. Also, FSKM-k is equivalent to SVM_RBF except for two outliers that probably resulted from a poor parameter choice of random grid search.

D. Resistance to attribute noise

Noise is an inherent characteristic in observational data that can degrade the performance of models [51]–[53]. Machine learning models usually model the noise in the predicted variable, for example, by using additive-noise statistical models. In supervised classification, the task is known as *class-noise* classification [51]–[53]. A less studied problem is for machine learning models to deal with input-noise, a.k.a, attribute-noise [51]–[53]. We hypothesize that the proposed FSKM are attribute-noise resistant kernel machines because of the non-singleton fuzzifier, which is an input noise-rejection filter [1].

1) Datasets and attribute-noise induction methodology: In this experiment, for each dataset from Table SM-2 we

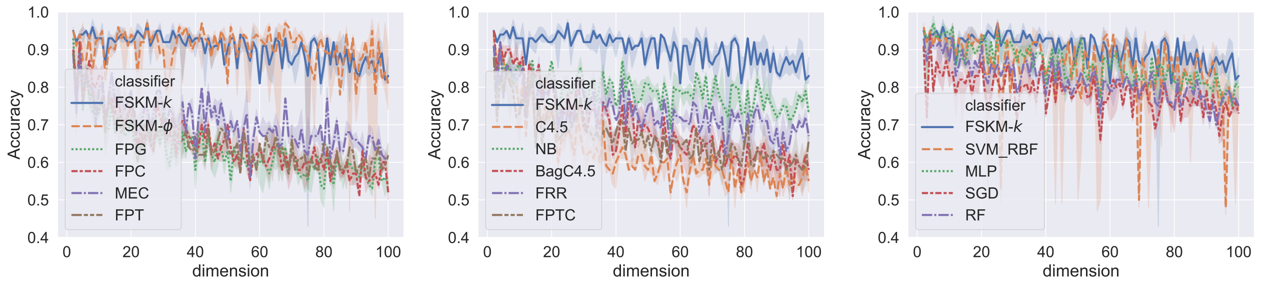

Fig. 5. Experimental results for the resistance to the curse of dimensionality. Each line represents the accuracy of a classifier as a function of the data dimension for a binary classification problem.

constructed eleven versions of them with different increasing attribute-noise levels. We applied the attribute noise injection methodology described in [51], i.e., for a noise level of $x\%$ and an arbitrary attribute, we randomly selected $x\%$ of the attribute values and replaced them by uniformly distributed random values within the attribute's domain values. We did that for all the attributes in the dataset, i.e., $x\%$ attribute values in the dataset were corrupted with noise. For this experiment, we used $0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50$ noise levels, which resulted in $26 \times 11 = 226$ attribute-noise datasets.

2) Experimental design and models: We used nested cross-validation as mentioned in Section V-B. Further, we used C4.5 Naïve Bayes and BagC4.5 as competing methods because they are the state-of-the-art for attribute-noise resistant machine learning models suggested by literature [51]–[53]. Table VIII shows the models used in this experiment.

TABLE VIII
CLASSIFIERES USED IN THE ATTRIBUTE-NOISE EXPERIMENT

Label	Classifier
FSKM-k	Sec IV-B
FSKM-ϕ	Sec IV-C
C4.5	C4.5 decision tree algorithm
BagC4.5	Bagging C4.5
NB	Gaussian Naive Bayes

3) Classifier comparison using Bayesian tests and discussion of results: Figures 6, SM-1 and 7 shows the Hierarchical Bayesian test results. The y-axis denotes probabilities, and the x-axis denotes the noise level. Each color point means the probability assigned by the Bayesian test for comparing two classifiers on all the datasets at a particular noise level. The dashed red line represents the threshold probability of 0.95. Given the twenty-six datasets, one can conclude that FSKM is better than another classifier with a probability threshold of 0.95 and rope=0 (no region of practical equivalence) if its color point is within the region above the dashed red line. On the contrary, color points below the dashed red line indicate that FSKM is not better than the other classifier, or no decision can be made based on the threshold of 0.95.

The left plot of Figure 6 shows the comparison results between FSKM-k vs. competing classifiers in terms of the probabilities: $P(\text{FSKM-}k \gg \text{clf})$, where *clf* denotes one of the classifiers: BagC4.5, C4.5, or, NB. From this result, we conclude that FSKM-k is better than C4.5 and NB classifiers on all the noise levels. Also, FSKM-k is better than BagC4.5 only at noise-level zero. The right plot of Figure 6 shows a similar analysis for the FSKM-ϕ classifier in terms of the probabilities: $P(\text{FSKM-}\phi \gg \text{clf})$. In this case, FSKM-$\phi$ is better than NB in the first seven noise levels and better than C4.5 at six noise levels. Figure SM-1 shows additional experiments where we conclude that FSKM-k is better than SVM_RBF, KNN, and MLP in all the eleven noise-levels; and better than FSKM-ϕ in all the noise levels greater than fifteen[25].

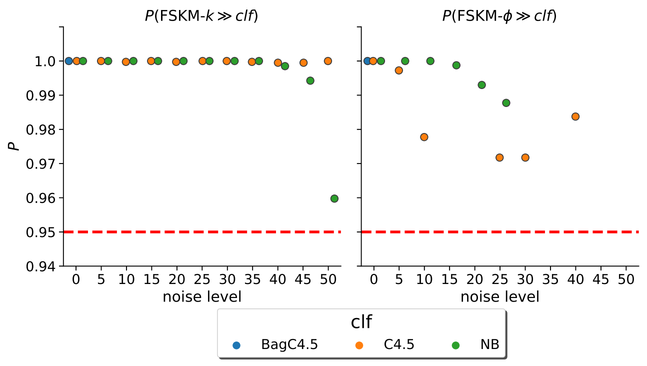

Fig. 6. Hierarchical Bayesian test results for the attribute-noise experiment. Each color point represents the probability given by the test quantifying if an FSKM classifier is better than competing classifiers on the attribute-noise classification task. The region above the dashed red line shows the decision zone based on a threshold of 0.95 probability value.

The left plot of Figure 7 shows the Bayesian test results in terms of the probability $P(\text{FRKM} \ll \text{Bag.C4.5})$ at rope zero — all the color points within the region above the dashed red line indicates the superiority of Bag.C4.5 classifier. We observe that Bag.C4.5 is better than FSKM-ϕ for eight noise levels and FSKM-k for seven noise levels. The right plot of Figure 7 shows the test results in term of probabilities $P(\text{FRKM} = \text{Bag.C4.5})$ at rope 0.05. Thus, FSKM-k is equivalent to Bag.C4.5 at noise levels (0,5,10,15,50). FSKM-ϕ and Bag.C4.5 are equivalents only at noise levels of (0,5). Our experiments confirm the reported results in [53] that Bag.C4.5 is the best classifier for attribute-noise; however, as we show in Figure SM-2 Bag.C4.5 is a costly procedure in terms of computational time. The median of the execution time (middle black line in boxplots) of Bag.C4.5 is almost two

[25]A similar analysis for FSKM-ϕ is included in the supplemental material.

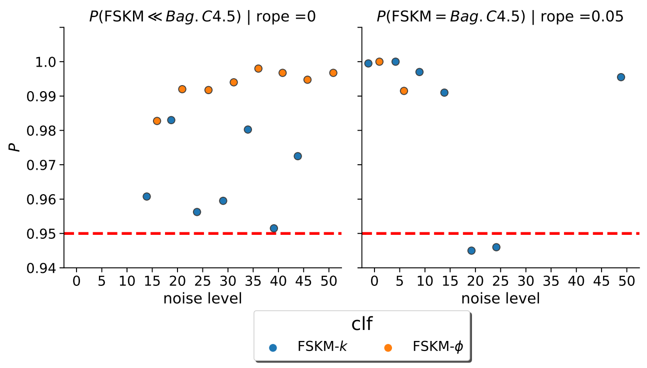

Fig. 7. Hierarchical Bayesian test results for the attribute-noise experiment. Each color point represents the probability given by the test quantifying if the BagC4.5 classifier is better (left plot) or equivalent at rope = 0.05 than the proposed FSKM classifiers. The region above the dashed red line shows the decision zone based on a threshold of 0.95 probability value.

orders of magnitude greater than the median of the execution time of FSKM procedures. We conclude that the proposed FSKM classifiers are attribute-noise resistant machines, have beaten most of the competitive classifiers at several noise levels, are equivalent to Bag.C4.5 at lower noise levels, and are less costly than Bag.C4.5 in terms of computational time[26]. Also, although FSKM-k and FSKM-ϕ have similar behavior at zero noise levels and are better than most of the competing classifiers at zero noise level (see the generalization power experiment in Section V-B), the FSKM-k classifier is more noise resistant than FSKM-ϕ for increasing attribute-noise levels (see additional experiments in Figure SM-1).

VI. Conclusions

We have presented the first step in studying the connection between fuzzy inference systems and kernel machines. For doing that, we used the characterization of the solution of regularized kernel machines given by the Representer Theorem and the fuzzy-basis-function (FBF) expansion of non-singleton Mamdani fuzzy systems (NSFSs). Based on that, we proposed two novel models: FSKM-k and FSKM-ϕ. Those models are kernel machines and fuzzy systems at the same time.

We experimentally verified some of the two-way connection claims. From the fuzzy systems side, we showed that they: 1) can be *trained by kernel algorithms*; 2) have more *resistance to the curse of dimensionality* than other fuzzy classifiers; 3) satisfy the Representer Theorem for kernel machines; 4) learn functions in RKHS, and 5) explicitly regularized the learned function. From the kernel perspective, we experimentally verified that kernel machines: 1) can be applied to fuzzy datasets; 2) can inherit the attribute-noise property of NSFS and; 3) can have IF-THEN rules by considering its memorized instances to be the antecedent part of IF-THEN rule antecedents of NSFSs.

As a future topic of research, we leave a more careful analysis of the role of the IF-THEN rules in the kernel machines realm. Open questions are: 1) how can those rules help with the interpretability/explainability of kernel machines? 2) How can those rules help set priors on the structure of the model for predicting small data?; 3) How could those rules help to improve the out-of-sample performance of kernel machines when the test distribution differs from the training distribution?

Finally, there are also exciting research questions regarding 1) the generalization of the studied connection to other types of fuzzy systems like Takagi-Sugeno-Kang or type-2 fuzzy systems; 2) the application of FSKM models in other tasks like regression and anomaly detection; 3) the use of the kernel structure beyond the arguments presented in this paper, e.g., kernel low-rank approximations for rule reduction 4) the connection with Bayesian approaches as Gaussian processes models; 5) the link between non-parametric statistical estimators based on non-negative kernels like KDE and fuzzy data.

[26]Figures SM-4 and SM-5 from the Supplemental Material show the complete set of experimental results for the twenty-six datasets with eleven attribute-noise levels.

References

[1] J. M. Mendel, *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions, 2nd Edition*, 2nd ed. Springer Publishing Company, Incorporated, 2017.

[2] A. Berlinet and C. Thomas-Agnan, *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011.

[3] B. Schölkopf, A. J. Smola, F. Bach *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[4] J. Shawe-Taylor, N. Cristianini *et al.*, *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[5] J. Guevara, J. M. Mendel, and R. Hirata, "Connections between fuzzy inference systems and kernel machines," in *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2020, pp. 1–8.

[6] G. C. Mouzouris and J. M. Mendel, "Nonsingleton fuzzy logic systems: theory and application," *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 1, pp. 56–71, 1997.

[7] J. Guevara, R. Hirata, and S. Canu, "Positive definite kernel functions on fuzzy sets," in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2014, pp. 439–446.

[8] ——, "Cross product kernels for fuzzy set similarity," in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2017, pp. 1–6.

[9] R. Martin-Clouaire, "Semantics and computation of the generalized modus ponens: The long paper," *International Journal of Approximate Reasoning*, vol. 3, no. 2, pp. 195–217, 1989.

[10] N. Aronszajn, "Theory of reproducing kernels," *Transactions of the American mathematical society*, vol. 68, no. 3, pp. 337–404, 1950.

[11] T. Gartner, *Kernels for structured data*. World Scientific, 2008, vol. 72.

[12] G. Shmueli *et al.*, "To explain or to predict?" *Statistical science*, vol. 25, no. 3, pp. 289–310, 2010.

[13] B. Kosko, "Fuzzy systems as universal approximators," *IEEE transactions on computers*, vol. 43, no. 11, pp. 1329–1333, 1994.

[14] L.-X. Wang, J. M. Mendel *et al.*, "Fuzzy basis functions, universal approximation, and orthogonal least-squares learning," *IEEE transactions on Neural Networks*, vol. 3, no. 5, pp. 807–814, 1992.

[15] L.-X. Wang, "Fuzzy systems are universal approximators," in *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*. IEEE, 1992, pp. 1163–1170.

[16] R. R. Yager and L. A. Zadeh, *An introduction to fuzzy logic applications in intelligent systems*. Springer Science & Business Media, 2012, vol. 165.

[17] H. Ying, *Fuzzy control and modeling: analytical foundations and applications*. IEEE press, 2000.

[18] Y. L. Sun and M. J. Er, "Hybrid fuzzy control of robotics systems," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 6, pp. 755–765, 2004.

[19] C. P. Chen, C.-Y. Zhang, L. Chen, and M. Gan, "Fuzzy restricted boltzmann machine for the enhancement of deep learning," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 6, pp. 2163–2173, 2015.

[20] L. A. Zadeh, "Fuzzy sets," in *Fuzzy sets, fuzzy logic, and fuzzy systems: selected papers by Lotfi A Zadeh*. World Scientific, 1996, pp. 394–432.

[21] D. Dubois and H. Prade, "The three semantics of fuzzy sets," *Fuzzy sets and systems*, vol. 90, no. 2, pp. 141–150, 1997.

[22] J. A. Goguen, "L-fuzzy sets," *Journal of mathematical analysis and applications*, vol. 18, no. 1, pp. 145–174, 1967.

[23] H. Bustince, E. Barrenechea, M. Pagola, J. Fernandez, Z. Xu, B. Bedregal, J. Montero, H. Hagras, F. Herrera, and B. De Baets, "A historical account of types of fuzzy sets and their relationships," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 1, pp. 179–194, 2015.

[24] E. H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Computer Architecture Letters*, vol. 26, no. 12, pp. 1182–1191, 1977.

[25] W. Van Leekwijck and E. E. Kerre, "Defuzzification: criteria and classification," *Fuzzy sets and systems*, vol. 108, no. 2, pp. 159–178, 1999.

[26] L.-X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on systems, man, and cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[27] Y. Chen and J. Z. Wang, "Support vector learning for fuzzy rule-based classification systems," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 716–728, 2003.

[28] J.-H. Chiang and P.-Y. Hao, "Support vector learning mechanism for fuzzy rule-based modeling: a new approach," *IEEE Transactions on Fuzzy systems*, vol. 12, no. 1, pp. 1–12, 2004.

[29] S.-M. Zhou and J. Q. Gan, "Constructing l2-svm-based fuzzy classifiers in high-dimensional space with automatic model selection and fuzzy rule ranking," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 3, pp. 398–409, 2007.

[30] C.-T. Lin, C.-M. Yeh, S.-F. Liang, J.-F. Chung, and N. Kumar, "Support-vector-based fuzzy neural network for pattern classification," *IEEE Transactions on Fuzzy Systems*, vol. 14, no. 1, pp. 31–41, 2006.

[31] H.-C. Huang, Y.-Y. Chuang, and C.-S. Chen, "Multiple kernel fuzzy clustering," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 1, pp. 120–134, 2011.

[32] X. Yang, G. Zhang, J. Lu, and J. Ma, "A kernel fuzzy c-means clustering-based fuzzy support vector machine algorithm for classification problems with outliers or noises," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 1, pp. 105–115, 2010.

[33] G. Heo and P. Gader, "Robust kernel discriminant analysis using fuzzy memberships," *Pattern recognition*, vol. 44, no. 3, pp. 716–723, 2011.

[34] B. Moser, "On the t-transitivity of kernels," *Fuzzy Sets and Systems*, vol. 157, no. 13, pp. 1787–1796, 2006.

[35] ——, "On representing and generating kernels by fuzzy equivalence relations." *Journal of machine learning research*, vol. 7, no. 12, 2006.

[36] J. Guevara, R. Hirata, and S. Canu, "Fuzzy set similarity using a distance-based kernel on fuzzy sets," 2015.

[37] C.-F. Juang, S.-H. Chiu, and S.-W. Chang, "A self-organizing ts-type fuzzy network with support vector learning and its application to classification problems," *IEEE transactions on Fuzzy Systems*, vol. 15, no. 5, pp. 998–1008, 2007.

[38] J. M. Łski, "On support vector regression machines with linguistic interpretation of the kernel matrix," *Fuzzy Sets and Systems*, vol. 157, no. 8, pp. 1092–1113, 2006.

[39] C. T. Wee and T. W. Wan, "Efsvm-fcm: Evolutionary fuzzy rule-based support vector machines classifier with fcm clustering," in *2008 IEEE International Conference on Fuzzy Systems (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 606–612.

[40] W.-Y. Cheng and C.-F. Juang, "An incremental support vector machine-trained ts-type fuzzy system for online classification problems," *Fuzzy Sets and Systems*, vol. 163, no. 1, pp. 24–44, 2011.

[41] G. C. Cawley and N. L. Talbot, "On over-fitting in model selection and subsequent selection bias in performance evaluation," *The Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.

[42] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2653–2688, 2017.

[43] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[44] I. Guyon, "Design of experiments of the nips 2003 variable selection benchmark," in *NIPS 2003 workshop on feature extraction and feature selection*, vol. 253, 2003.

[45] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[46] S. A. Davidsen, E. Sreedevi, and M. Padmavathamma, "Local and global genetic fuzzy pattern classifiers," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*. Springer, 2015, pp. 55–69.

[47] C. Stoean, R. Stoean, M. Preuss, and D. Dumitrescu, "Diabetes diagnosis through the means of a multimodal evolutionary algorithm," in *Proceedings of the First East European Conference on Health Care Modelling and Computation-HCMC*, 2005, pp. 277–289.

[48] R. Senge and E. Hüllermeier, "Top-down induction of fuzzy pattern trees," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 241–252, 2010.

[49] S. K. Meher, "A new fuzzy supervised classification method based on aggregation operator," in *2007 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System*. IEEE, 2007, pp. 876–882.

[50] Z. Huang, T. D. Gedeon, and M. Nikravesh, "Pattern trees induction: A new machine learning method," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 4, pp. 958–970, 2008.

[51] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artificial intelligence review*, vol. 22, no. 3, pp. 177–210, 2004.

[52] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artificial intelligence review*, vol. 33, no. 4, pp. 275–306, 2010.

[53] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Tackling the problem of classification with noisy data using multiple classifier systems: Analysis of the performance and robustness," *Information Sciences*, vol. 247, pp. 1–20, 2013.

Jorge Guevara earned a Ph.D. degree in Computer Science from the University of Sao Paulo, Brazil. He is a Research Scientist at IBM Research. He actively works in three research lines: i) hybrid machine learning models based on kernel methods and real-valued logic; ii) predictive modeling for oil and gas; ii) AI-based Weather Generators for Climate Applications — He holds several papers and patents in those areas.

Jerry M. Mendel (LF'04) received the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, NY. Currently, he is Emeritus Professor of Electrical Engineering at the University of Southern California in Los Angeles. He has published over 580 technical papers and is author and/or co-author of 12 books. He is a Life Fellow of the IEEE, a Distinguished Member of the IEEE Control Systems Society, and a Fellow of the International Fuzzy Systems Association. He was a member of the Administrative Committee of the IEEE Computational Intelligence Society for nine years, and Chairman of its Fuzzy Systems Technical Committee and the Computing With Words Task Force of that TC. Among his awards are four IEEE Transactions Best/Outstanding paper awards, a 1984 IEEE Centennial Medal, an IEEE Third Millenium Medal, a Fuzzy Systems Pioneer Award (2008) from the IEEE Computational Intelligence Society, and the IEEE Lotfi A. Zadeh Pioneer Award (2021). His present research interests include: type-2 fuzzy logic systems and XAI. He has more than 58,700 citations on Google Scholar.

Roberto Hirata Jr. is an associate professor at the Computer Science Department - University of São Paulo. He has a PhD degree in Computer Science (2001). his research is in Computer Vision and Machine Learning. He was also a professor for the SENAC College of Computer Science. He advised 8 doctorates and 16 masters and he is currently a principal investigator in two large projects (FAPESP - 2014/50937-1 - Future Internet for Smart Cities; and FAPESP/CNPq - 2014/465446-0 - INCT InterSCity - Enabling the Future Internet for Smart Cities).