Gath-Geva Specification and Genetic Generalization of Takagi-Sugeno-Kang Fuzzy Models

Martin Berchtold, Till Riedel, Christian Decker
TecO, University of Karlsruhe

Kristof van Laerhoven
MIS, TU Darmstadt

***Abstract*—This paper introduces a fuzzy inference system, based on the Takagi-Sugeno-Kang model, to achieve efficient and reliable classification in the domain of ubiquitous computing, and in particular for smart or context-aware, sensor-augmented devices. As these are typically deployed in unpredictable environments and have a large amount of correlated sensor data, we propose to use a Gath-Geva clustering specification as well as a genetic algorithm approach to improve the model's robustness. Experiments on data from such a sensor-augmented device show that accuracy is boosted from 83% to 97% with these optimizations under normal conditions, and for more challenging data from 54% to 79%.**

I. Introduction

Context-awareness and smart devices have always accompanied the vision of ubiquitous computing since its postulation by Weiser [1]. Motivated by our work in this area, we identify two specific challenges in fuzzy classification that are typical (but not limited) to this application domain.

The first problem which we address in this paper concerns the specification of fuzzy classifiers. While classifying data using fuzzy inference systems (FIS) is also common in other areas, the specification of such systems meets special challenges. Because ubiquitous computing research focusses on unpredictable environments and the input features are manifolds, it is difficult to specify classification from prior domain knowledge. Furthermore, the final classifiers are targeted for execution on resource-constrained embedded devices like the AwarePen artifact [2], which we chose for this paper's case study. This condition makes automatically constructed Takagi-Sugeno-Kang Fuzzy Models a perfect match: Using annotated data, they can be trained and adapted offline, and executed efficiently on the smart item's embedded platform. In previous work, we successfully applied automatically constructed TSK-FIS in this application domain using multivariate subtractive clustering for construction [3]. One challenge is the fact that many classifiers depend on the correlation of features in the input, with high dimensionality and often a huge variance. We will show that non-covariant clustering will result in a high number of cluster to cover the input space, and consequently rule executions at runtime, which is problematic both for execution on smart items, as well for interpretability. A Gustafson-Kessel clustering algorithm [4] would provide covariant membership functions, but the clustering has some disadvantages: An adaption to different cluster densities is not possible, a constant cluster volume is recommended, or the volume needs to be defined by hand. In this paper we approach this problem by proposing a novel construction algorithm for TSK models using a Gath-Geva clustering. A modified Gath-Geva clustering was already proposed in Abonyi et al [5]. Their modified clustering algorithm does not rely on the transformation of the input domain, thus preserving the interpretability of the resulting TSK model. They do not take the correlation of the inputs into account, which is not preferable in our case. We need to capture the correlation of the inputs through the multidimensional covariant membership functions for our particular classification problem. Having a interpretable model on input dimension level is in our case not necessary, just a interpretation of rules per class is needed. The second of the before mentioned challenges in using automatically constructed FIS in the field of ubiquitous computing is, that systems tend to overly adapt to the training data. Smart artifacts like AwarePen are typically used by many different users in a number of different environments. Training, however, requires annotated data and can only be done involving a limited number of people and is often done in few controlled environments for pratical reasons. We argue that this shortage of adequate training data, which is intrinsic to the ubiquitous computing domain, can be overcome by a generalization of the FIS model. As the second part of our construction process, we propose a genetic algorithm approach to remove conflicting over-specifications by a generalization of the TSK FIS. Already in the work of Setnes et al [6] a genetic algorithm is used to optimize the antecedent and the consequent variables of TSK models. They use a supervised fuzzy clustering algorithm to obtain the rules of the model. Further work of Roubos and Setnes [7] again adopts a genetic algorithm for parameter tuning and a similarity measure to simplify the rules of the FIS. A tuning of variables is in our case not necessary, since the used Gath-Geva clustering achieves a good approximation of the covariant training data. The genetic optimization in our case is done upon a static topology of an TSK-FIS without tuning the variables of the fuzzy system. The approach of Chou et al [8] deals with incomplete data via a combination of fuzzy c-means (FCM) and Dempster-Shafer theory. Their aim is to classify data records with missing values, therefore they use the FCM to get an initial degree of belief for complete data, and use the Dempster-Shafer theory to make a final decision of the class to which the incomplete data should belong to. This approach is not optimal in our case, since at construction time of the TSK-FIS the data for classification is complete. In the further use of the classifier we deal with data that can

1-4244-2384-2/08/$20.00

be in nature contradicting to the classifier rules, or at least to some of the rules dimensions. Therefore, a classifier that can be generalized at runtime is required, so that unknown data does not remain contradicting, without changing the topology of the classifier or adapting to the new data. Adapting the classifier onto the new data could lead to suboptimal classification abilities for the normal data

First, we describe our proposed Fuzzy Inference System (FIS) for classification in section II which is based on the Takagi-Sugeno-Kang FIS. This is followed in section III with the description of our specification approach, and two application examples in section IV, which serve as evaluation of our model. The last section sums up our conclusions and briefly describes the future steps in this research.

II. Fuzzy Inference System for Classification

A. Takagi-Sugeno-Kang-FIS

Takagi, Sugeno and Kang [9][10] (TSK) fuzzy inference systems are fuzzy rule-based structures, which are especially suited for automated construction. Within a TSK-FIS, the consequence of the implication is not a functional membership to a fuzzy set, but a constant or linear function. The consequence of the rule j depends on the input of the FIS:

$$f_j(\overrightarrow{\mathbf{v}}_t) := \sum_{i=1}^{n} a_{ij} v_i + a_{(n+1)j} \tag{1}$$

The linguistic equivalent of a rule j is formulated accordingly:

$$\text{IF } \mu_{1j}(v_1) \text{ AND } \mu_{2j}(v_2) \text{ AND .. AND } \mu_{nj}(v_n) \text{ THEN } f_j(\overrightarrow{\mathbf{v}}_t) \tag{2}$$

The membership functions of the rule are non-linear Gaussian functions. The antecedent part of the rule j determines the weight w_j accordingly:

$$w_j(\overrightarrow{\mathbf{v}}_t) := \prod_{i=1}^{n} \mu_{ij}(v_i) \tag{3}$$

The projection from input $\overrightarrow{\mathbf{v}}_t := (v_1, v_2, .., v_n)$ onto the classifiable one-dimensional set is a weighted sum average, which is a combination of fuzzy reasoning and defuzzification. The weighted sum average is calculated according to the rules $j = 1, .., m$ as follows:

$$\mathbf{S}(\overrightarrow{\mathbf{v}}_t) := \frac{\sum_{j=1}^{c} w_j(\overrightarrow{\mathbf{v}}_t) f_j(\overrightarrow{\mathbf{v}}_t)}{\sum_{j=1}^{c} w_j(\overrightarrow{\mathbf{v}}_t)} \tag{4}$$

B. TSK-FIS with Covariant Membership Functions

In ubiquitous computing we are mostly dealing with highly correlated data. With multivariate membership functions (MF) the data cannot be covered sufficiently, therefore more functions are needed. Since multivariate MF cannot adapt to the shape of covariant data, the data is not accurately represented by the MF, and therefore a separation of different classes results in a bigger error. On the other hand, multidimensional covariant MF's are not as intuitively interpretable as seperate multivariate MF's. In our case only an interpretation of the model on rule level is needed.

We opt for a covariant MF, which results in less rules and smaller error. The covariant MF is defined accordingly:

$$\mu_j(\overrightarrow{\mathbf{v}}_t) := e^{-\frac{1}{2}(\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j)\Sigma_j^{-1}(\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j)^T} \tag{5}$$

With the covariant MF the whole input of each rule is handled by one MF, which results in a simplified rule:

$$\text{IF } \mu_j(\overrightarrow{\mathbf{v}}_t) \text{ THEN } f_j(\overrightarrow{\mathbf{v}}_t) \tag{6}$$

The whole antecedent part of each rule was multiplied with the usual TSK-FIS (eqn. 3) to get the respective weight, but with the covariant MF's the function is already the weight. The resulting formula for the covariant TSK-FIS is defined, as follows:

$$\mathbf{S}(\overrightarrow{\mathbf{v}}_t) := \frac{\sum_{j=1}^{m} \mu_j(\overrightarrow{\mathbf{v}}_t) f_j(\overrightarrow{\mathbf{v}}_t)}{\sum_{j=1}^{m} \mu_j(\overrightarrow{\mathbf{v}}_t)} \tag{7}$$

In further evaluations and analysis of this paper, we will refer to the covariant TSK-FIS defined in equation 7.

C. Fuzzy Classification

The outcome of the TSK-FIS mapping needs to be assigned to one of the classes the projection should result in. This assignment is done fuzzy, so the result is not only a class identifier, but also a membership identifying the confidence of the classification process. Each class identifier is interpreted as a triangular shaped fuzzy number. The mean of the fuzzy number is the identifier itself, with the highest membership of one. The crisp decision, which identifier is the mapping outcome, is carried out based on the highest membership to one of the class identifiers. The overall output of the FIS mapping and the classification is a tuple $(\mathcal{C}, \mu_{\mathcal{C}})$ of class identifier, and the membership to it.

III. Machine Learning Algorithms

A. Specification of TSK-FIS with Covariant Membership Functions

For the automatic specification of the fuzzy inference system a combination of clustering algorithms and linear regression was used. The clustering was done upon a training data set, which was separated according to the class each sample belongs to. Due to this class specific clustering, every rule resulting from each cluster is clearly associable with the class it is resulting from. This was done on the one side to make the resulting FIS interpretable, and on the other side, to optimize on class specific features.

1) Subtractive Clustering: The biggest advantages of Gath-Geva clustering [11] are the adaption on covariant shapes and various cluster densities, which are not easy to realize with Gustafson-Kessel clustering [4]. Although, two problems with the clustering algorithm are arising from the usage in an automatic FIS identification: First, the algorithm needs to know the amount of clusters, and second, for a random initialization it might not always converge to the optimal cluster centers. Thus, a method is needed to determine the initial clusters centers. Mountain clustering [12] may be used for initialization, but is highly dependent on the grid structure.

We opt for a subtractive clustering [13] instead. This clustering estimates every data point as a possible cluster center, so there are no prior specifications. Chiu [14] gives a description of the parameters that the subtractive clustering needs for good cluster determination. We use subtractive clustering to determine a set of cluster centers, that provide an upper bound for the Gath-Geva clustering.

2) Initial Cluster Determination: The subtractive clustering [13] can give an upper bound on the amount of clusters, however, because it cannot adapt to covariant cluster shapes, it needs many fuzzy cluster functions to adapt to the data. An example is given in figure 1 on the left side for the subtractive clustering and on the right for the Gath-Geva clustering. A criterion is now needed to summarize multivariate clusters, which can be covered through one covariant one.

The *Partition Coefficient (PC)*[15] measures the amount of overlapping between clusters. Since the overlapping does not directly indicate the correlation of the clusters, this criterion can only be partly used. The same applies for the *Classification Entropy (CE)*, which is only a slight variation of the *PC*.

The *Partition Index (PI)* indicates the compactness and separation of clusters. A Gath-Geva cluster has different size and density characteristics, and therefore different nature in separation, than the subtractive clustering, thus if the *PI* indicating a good partitioning, this might not be the case for a covariant cluster shape. Due to the similarity of the *Separation Index (SI)* to the *PI*, this is also not practical in our case.

Other validity measures for clustering results, e.g. *Dunn's Index (DI)* and *Alternative Dunn Index (ADI)*, also aim on clustering specific similarities or dissimilarities, which cannot easily be used for parametrization of different cluster algorithms.

After evaluating the different validity measures on our problem, we ended up with a different approach. Our method uses the upper bound of cluster numbers derived through a subtractive clustering as starting point, and than successively reduces the amount, until a best classification through the resulting covariant TSK-FIS is reached. Similar approaches can be found in [16] and [17].

3) Gath-Geva Clustering: Gath and Geva [11] generalize the maximum likelihood estimation for the fuzzy clustering. They assume, that the normal distribution N_j, with the expected value for the cluster center $\overrightarrow{\mathbf{m}}_j$, the covariance matrix Σ_j, and the a-priory probability P_j are used to generate the data $\overrightarrow{\mathbf{v}}_t \in \mathbf{V}_T$.

For initialization of the clustering, a set of cluster centers $M = \{\overrightarrow{\mathbf{m}}_1, ..\overrightarrow{\mathbf{m}}_c\}$ needs to be estimated, in our case we use a subset of the centers found through the subtractive clustering. The number of clusters c has an upper bound found through the subtractive clustering, which is initially used. The initial memberships are calculated according to the fuzzy c-means clustering through a Euclidian distance, as follows:

$$\mu_j^{(1)}(\overrightarrow{\mathbf{v}}_t) = \frac{1}{\|\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j\|} \tag{8}$$

in further iterations $l = 2, 3..$ of the algorithm, the membership is calculated according to equation 5. First step of the clustering algorithm is to calculate the cluster centers $\overrightarrow{\mathbf{m}}_j$, as follows:

$$\overrightarrow{\mathbf{m}}_j^{(l)} = \frac{\sum_{t=1}^m \mu_j^{(l-1)}(\overrightarrow{\mathbf{v}}_t)\overrightarrow{\mathbf{v}}_t}{\sum_{t=1}^m \mu_j^{(l-1)}(\overrightarrow{\mathbf{v}}_t)} \tag{9}$$

Second step is to determine the covariance matrix, which is needed to calculate the distance measurement. The covariance matrix is calculated as follows:

$$\Sigma_j^{(l)} = \frac{\sum_{t=1}^m (\mu_j^{(l-1)}(\overrightarrow{\mathbf{v}}_t))^2 (\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j^{(l)})^T (\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j^{(l)})}{\sum_{t=1}^m (\mu_j^{(l-1)}(\overrightarrow{\mathbf{v}}_t))^2} \tag{10}$$

The algorithm employs a distance measurement based on the fuzzy maximum likelihood estimates, proposed by Bezdek and Dunn [15]:

$$D_{jt}(\overrightarrow{\mathbf{v}}_t, \overrightarrow{\mathbf{m}}_j) = \frac{\sqrt{|\Sigma_j|}}{P_j^{(l)}} e^{-\frac{1}{2}(\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j^{(l)})\Sigma_j^{-1}(\overrightarrow{\mathbf{v}}_t - \overrightarrow{\mathbf{m}}_j^{(l)})^T} \tag{11}$$

The a-priory probability is:

$$P_j^{(l)} = \frac{\sum_{t=1}^m \mu_j^{(l-1)}(\overrightarrow{\mathbf{v}}_t)}{\sum_{t=1}^m \sum_{l=1}^c \mu_l^{(l-1)}(\overrightarrow{\mathbf{v}}_t)} \tag{12}$$

The steps 1 and 2 of the algorithm get now repeated until iteration l is reached where $\sum_{j=1}^c |\overrightarrow{\mathbf{m}}_j^{(l)} - \overrightarrow{\mathbf{m}}_j^{(l-1)}| < \epsilon$.

4) Estimation of Consequence Parameters: The estimation of the consequence parameters is done via an ordinary least-squares method. For each rule the consequence parameters get calculated separately by minimizing the mean squared error:

$$\mathcal{E}(\mathbf{V}_T) = \sum_{t=1}^m (u_t - \mathbf{S}(\overrightarrow{\mathbf{v}}_t))^2 = \sum_{t=1}^m (u_t - \sum_{k=1}^{(n+1)c} b_{tk} a_k)^2 \tag{13}$$

The error gets minimized for each parameter of the consequence by setting its gradient to zero:

$$\frac{\partial \mathcal{E}}{\partial a_l} = \sum_{t=1}^m 2(u_t - \sum_{k=1}^{(n+1)c} b_{tk} a_k)(-b_{tl}) = 0 \tag{14}$$

Here u_t is the wanted outcome for input vector $\overrightarrow{\mathbf{v}}_t$, a_j is element of the vector $\overrightarrow{\mathbf{a}}$, and b_{tj} is element of matrix B_T, which is calculated as follows:

$$b_{tk} = \frac{\mu_j(\overrightarrow{\mathbf{v}}_t)}{\sum_{j=1}^c \mu_j(\overrightarrow{\mathbf{v}}_t)} v_{kt} \tag{15}$$

, where $k = 1, .., (n+1), .., (n+1)c$, the normalized weight of each rule j is repeated $(n+1)$ times, and $v_{((n+1)\cdot j)t} = 1$.

The parameters of the consequence part of each rule are concatenated in one vector, as follows:

$$\begin{aligned} \overrightarrow{\mathbf{a}} &= (\overbrace{a_{11}, .., a_{(n+1)1}}^{\text{consequence param. rule 1}}, \quad .. \quad , \overbrace{a_{1c}, .., a_{(n+1)c}}^{\text{consequence param. rule c}}) \\ &= (a_1, .., a_{(n+1)}, \quad .. \quad , a_{((n+2)\cdot(c-1))}, .., a_{((n+1)\cdot c)}) \end{aligned} \tag{16}$$

Solving the equation 14 for all parameters a_j leads to the following linear equation:

$$\begin{aligned} 2(B_T \overrightarrow{\mathbf{a}}^T - \mathbf{u})^T B_T &= 0 \\ \overrightarrow{\mathbf{a}}^T &= \left((B_T^T B_T)^{-1} B_T^T\right) \mathbf{u} \end{aligned} \tag{17}$$

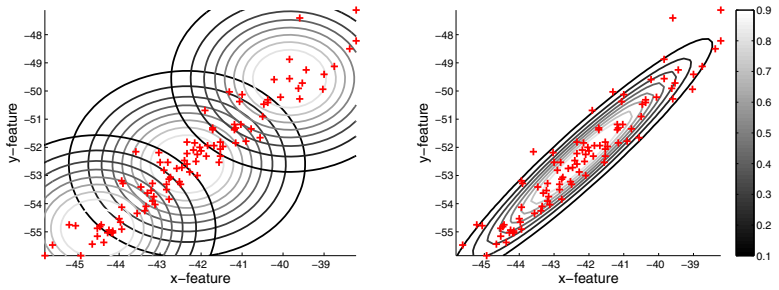

Fig. 1. Found fuzzy clusters on covariant data for subtractive clustering (left) and Gath-Geva clustering (right)

B. Generalization of Covariant TSK-FIS

The biggest problem in building classifiers in many ubiquitous computing applications is the lack of sufficient data at the time of construction and training. A mechanism is required to dynamically adapt the classification to unknown input data without changing the topology of the classifier.

We propose a method for the covariant TSK-FIS previously introduced, where individually on each rule is decided which of its inputs is relevant for classifying the new data and which inputs are producing confusion.

Since the combination of inputs n and rules c results in a search problem of $O(2^{n \cdot c})$, an evolutionary optimization method was chosen.

1) Bit Vector FIS Modification: A representation of the FIS is needed that specifies the dimensionality of each rule, which is in the current system either active or inactive. The two states to be specified allow the representation to be a bit vector, which is due to its efficient memory usage especially easy to store and process on platforms with limited resources. The described genetic algorithm optimization also works immediately with bit vectors.

First, a function is defined that maps the general FIS $\mathbf{S}$ according to the bit specification onto a modified FIS $\mathbf{S}'$ with modified rule dimensionality. The mapping function $\mathcal{I}$ is rather an interpretation, than a modification of the FIS, since it does only temporarily 'switch' the inputs of the rules on or off, and does not modify the topology of the FIS. The interpretation function $\mathcal{I}$ is defined, as follows:

$$\mathcal{I} : \left\{ \begin{array}{ccc} \mathcal{F}_{FIS}(\mathbb{R}^n, \mathbb{R}) & \longrightarrow & \mathcal{F}_{FIS}(\mathbb{R}^n, \mathbb{R}) \\ \mathbf{S}(\overrightarrow{\mathbf{v}}_t) & \longmapsto & \mathcal{I}(bit_{\mathbf{S}}, \mathbf{S}(\overrightarrow{\mathbf{v}}_t)) = \mathbf{S}'(\overrightarrow{\mathbf{v}}_t) \end{array} \right. \tag{18}$$

The interpretation $\mathcal{I}$ maps from the space of FIS functions $\mathcal{F}_{FIS}$ onto the same space, where $\mathcal{F}_{FIS}(\mathbb{R}^n, \mathbb{R})$ is a set of FIS functions mapping from $\mathbb{R}^n$ onto $\mathbb{R}$.

The bit vector specifies the inputs of the rules, which is described through the following example:

$$\begin{array}{cccc} & \overbrace{\text{rule } 1} & \ldots & \overbrace{\text{rule } m} \\ bit_{\mathbf{S}} = & (1,0,1,..,0,1, & \ldots & ,0,1,0,..,1,0) \\ & (v_1,-,v_3,..,-,v_n) & \ldots & (-,v_2,-,..,v_{n-1},-) \\ & w'_1(v_1,v_3,..,v_n) & \ldots & w'_m(v_2,..,v_{n-1}) \\ & f'_1(v_1,v_3,..,v_n) & \ldots & f'_m(v_2,..,v_{n-1}) \end{array} \tag{19}$$

There are two ways to modify the rules: One is to reduce the dimensionality of the antecedent membership function and the consequence, the other is to merely adjust the antecedent parts. As can be seen in the evaluation, the modification of both results in some cases in slightly better classification rate, but the mean squared error (MSE) has also been seen to increase.

2) Genetic Algorithm: Genetic Algorithms belong to the group of evolutionary methods. Originally, they were devised for optimization with the ability to solve non-linear non-quadratic optimization problems [18][19]. The aim is to find a minimum for a fitness function through modifying the individuals, which are specified through their genomes. From each generation a subset of the individuals (population) with the best fitness is selected for the next generation. A subset of the best individuals also get mutated and recombined. The size of each generation is constant. The bigger the search space, the more individuals are needed.

In our optimization, the fitness is chosen to be the percentage of false classifications, the less the better. The percentage is calculated upon a check set, which is independent of the training set. A fitness based on the MSE is possible, but mostly leads to less accurate classifications, because outliers increase the MSE, but not necessary the classification accuracy.

The representation of the genome as bit vector is especially efficient for optimization. Would the genome consist of real values, then the variations, and with them the search space, would be increasing to infinity, whereas the bit variations are limited. The mutations of the individuals are within a certain hamming distance to the original genome.

The algorithm stops if, for a certain number of generations, the fitness of the best individual does not improve anymore.

IV. Application Examples

The data for our examples were collected with a typical ubiquitous and pervasive artifact, the AwarePen [2][3]. This artifact consists of a wireless communication device, a digital signal processor, and a 3-dimensional acceleration sensor. It is used to detect several states a pen can be situated in, e.g. 'writing with', 'lying on desk', 'point with', 'in pocket while walking', etc. The input of the classifier is the variance, and the mean over a window length L of every axis of acceleration, which results in 6 inputs. We now show how the covariance TSK-FIS reduces the classification error compared to the

multivariate one, and how the genetic optimization reduces the error for unknown data.

A. Example 1: Pen Acceleration Data - 3 Classes

In the first example we show the performance of our approach for a classifier that has three target classes. These classes are: 'lying on desk' (1), 'writing' (2), and 'pointing at slide' (3). The genetic algorithm's optimization is used to reduce the classification error for data that differs from the training data. Differences in the data for the 'lying on desk' is introduced by having a cell phone ringing next to the pen, thus resulting in slightly different motion data. The training data for the 'writing' class was recorded whilst writing on a desk, whereas the unknown data is resulting from writing on a vertical white board. Here, the mean values of the inputs are different, since the orientation of the pen has changed dramatically.

Upon the training data set of 900 data pairs, with about 300 per each of the three classes, the system was generated with the previously introduced method. The percentage of correct classifications for this training data set of the resulting covariant TSK-FIS was 96%, and for an independent test set (no unknown data) 94%. Another data set, called check data set, was used to optimize the system with the genetic algorithm. Half of the check data with 900 samples consisted of similar data to the test set, and half of it was the previously described unknown data. The classification accuracy before optimization was 80%. After selecting the best result over several runs of the genetic algorithm, the classification accuracy was improved to up to 99%. To test the optimization a test set (also about 500 samples) was used, different to the previously used sets, which consisted of training set similar and unknown data. The accuracy before optimization was 83% and 97% after. The confusion matrices for before (table I) and after (table II) optimization show how the different classes have improved, and where the false classifications are situated.

designated classes	classified onto class		
	1	2	3
1	68.7179	23.0769	8.2051
2	0	91.7874	8.2126
3	0	6.1224	93.8776

TABLE I
CONFUSION MATRIX FOR CHECK DATA (83%, MSE 0.5819) OF NON OPTIMIZED FIS

designated classes	classified onto class		
	1	2	3
1	100.0000	0	0
2	0.4831	93.2367	6.2802
2	3.0612	0	96.9388

TABLE II
CONFUSION MATRIX FOR CHECK DATA (97%, MSE 0.5819) OF OPTIMIZED FIS

B. Example 2: Pen Acceleration Data - 5 Classes

In the second example, we want to show that our approach also works for more classes and more unrelated check data than in the previous example. This is an upper limit of our optimization approach, but still gives reasonable results. The classes are: 'lying on desk' (1), 'writing' (2), 'pointing at slide' (3), 'pen in trouser's pocket whilst sitting' (4), and 'pen in trouser's pocket whilst standing' (5).

For all these target classes, extra motion patterns were introduced during the test dataset to evaluate for robustness of our approach. For the 'lying on desk' class, the new data contains noisy data from a ringing cellphone next to the pen, and the desk it is lying on being bumped into. The class 'writing' is constructed on 'writing on desk' data, and is optimized also for 'writing on white board'. A difference to the 'pointing at slide' class compared to the training set is introduced by turning the pen around and pointing with the front end at the slide. The 'pen in the trouser's pocket whilst standing' class has unknown data from a different and more active user, than the user in the training data. The last class 'pen in the trouser's pocket whilst sitting' is the most challenging one, since the user was once sitting on the edge of the chair, having her legs in a different angle, and another time the user is nervously moving her legs around. The automatically constructed covariant TSK-FIS classifiers upon the training set of 546 data pairs, which are less pairs for more classes than in the previous example, the resulting FIS was consisting of 18 rules, which was 6 less than the upper bound from the subtractive clustering. The accuracy for the training data set was 99% and for an independent but similar (no unknown data) test set was 90%. This high classification accuracy for fewer rules shows good coverage through the covariant membership functions. Detailed results can be seen for the test set in the confusion matrix of table III. The classification accuracy of 54% on the check data for this non-optimized covariant TSK-FIS is far from useful. The genetic optimization has therefore to improve the classifier substantially to get reasonable classification results. The bit vector genome is 108 bits long, which results in an increased search space for the optimization algorithm. After optimization, the accuracy for the check set is 78% and for the test set for optimization validation 75%. While improving the classification for the new data, the classification accuracy for the old test set (no unknown data) needs to

designated classes	classified onto class				
	1	2	3	4	5
1	89.4737	10.5263	0	0	0
2	0	98.4848	1.5152	0	0
3	0	0	100.0000	0	0
4	0	12.5000	31.2500	56.2500	0
5	0	0	0	0	100.0000

TABLE III
CONFUSION MATRIX FOR TEST DATA (90%, MSE 0.5819, 273 SAMPLES) OF CONSTRUCTED FIS WITHOUT OPTIMIZATION

stay the same, or at least worsen just a bit, which is even improved with an accuracy of 91% (before optim. 90%). The genetic optimization was done for both, the antecedent and the consequence part of each rule. The confusion matrices for both sets can be found in table IV and V.

designated classes	classified onto class				
	1	2	3	4	5
1	98.2301	1.7699	0	0	0
2	0	92.3077	7.6923	0	0
3	7.0312	60.9375	31.2500	0.7812	0
4	0	0	0	98.9362	1.0638
5	0	3.9216	19.6078	0	76.4706

TABLE IV
CONFUSION MATRIX FOR CHECK DATA (78%, MSE 3.3464, 554 SAMPLES) OF OPTIMIZED FIS (ANTECEDENT+CONSEQUENCE)

designated classes	classified onto class				
	1	2	3	4	5
1	95.4545	2.7273	1.8182	0	0
2	0.9091	84.5455	10.9091	3.6364	0
3	9.8361	57.3770	32.7869	0	0
4	0	0	0	98.2143	1.7857
5	0	4.0000	28.0000	0	68.0000

TABLE V
CONFUSION MATRIX FOR TEST DATA (75%, MSE 4.0262, 554 SAMPLES) OF OPTIMIZED FIS (ANTECEDENT+CONSEQUENCE)

For an optimization, only on the antecedent part of the rules, the classification accuracy is slightly improved to up to 79% (MSE 1.3127) for the check set, 79% (MSE 5.9081) for the test set, and the test set for construction of the FIS 86% (before optim. 90%). The confusion matrix for this classifier is in table VI.

designated classes	classified onto class				
	1	2	3	4	5
1	88.1818	7.2727	4.5455	0	0
2	2.7273	94.5455	0.9091	1.8182	0
3	3.2787	68.8525	23.7705	3.2787	0.8197
4	0.8929	0	1.7857	97.3214	0
5	0	2.0000	0	0	98.0000

TABLE VI
CONFUSION MATRIX FOR TEST DATA (79%, MSE 5.9081, 554 SAMPLE) OF OPTIMIZED FIS (ANTECEDENT)

V. CONCLUSION

Two optimizations have been proposed to the Takagi-Sugeno-Kang model, to enhance its performance in particular in ubiquitous computing applications. These are challenging since they present the classifier with unpredictable environments of deployment, are implemented on platforms with limited resources, and have typically highly correlated input data. Data from one such project, a sensor-augmented writing pen, was used as a case study. We argue for the usage of a Gath-Geva-clustering specification, as well as an evolutionary algorithm approach to improve the model's robustness, while keeping the classifier resource-efficient enough at runtime. Evaluation on the data from the case study show that accuracy has improved from 83% to 97%, with these optimizations under normal conditions, and for more challenging data from 54% to 79%.

ACKNOWLEDGMENT

The work presented in this paper was funded by the German Ministry for Education and Research (BMBF) through the project LoCostix.

REFERENCES

[1] M. Weiser, "The computer of the 21st century," *Sci. American*, 1991.
[2] M. Berchtold, C. Decker, T. Riedel, T. Zimmer, and M. Beigl, "Using a context quality measure for impr. smart appliances," *IWSAWC*, 2007.
[3] M. Berchtold, T. Riedel, M. Beigel, and C. Decker, "Awarepen - classfication probability and fuzziness in a context aware application," *LNCS Transactions on Ubiquitous Intelligence and Computing*, 2008.
[4] E. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," *IEEE CDC, San Diego, California, pp 761-766*.
[5] J. Abonyi, R. Babuska, and F. Szeifert, "Modified gath-geva fuzzy clustering for identification of takagi-sugeno fuzzy models," *Systems, Man and Cybernetics, 2002, vol. 32 pp. 612-621*, 2002.
[6] M. Setnes and H. Roubos, "Transparent fuzzy modeling using fuzzy clustering and ga's," *North American Fuzzy Inf. Proc. Society*, 1999.
[7] H. Roubos and M. Setnes, "Compact fuzzy models through complexity reduction and evolutionary optimization," *In Proc. of the Ninth IEEE International Conference on Fuzzy Systems*, 2000.
[8] T.-S. Chou, K. K. Yen, L. An, N. Pissinou, and K. Makki, "Fuzzy belief pattern classification of incomplete data," *IEEE International Conference on Systems, Man and Cybernetics*, 2007.
[9] T. Tagaki and M. Sugeno, "Fuzzy identification of systems and its application to modelling and control," *Syst., Man and Cybernetics*, 1985.
[10] M. Sugeno and G. Kang, "Structure identification of fuzzy model," *Fuzzy Sets and Systems, 1988, vol 26(1), pp 15-33*, 1988.
[11] I. Gath and A. B. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 11(7), pp 773-781*, 1989.
[12] R. Yager and D. Filer, "Generation of fuzzy rules by mountain clustering," *Journal on Intelligent Fuzzy Systems, vol 2, pp 209-219*, 1994.
[13] S. Chiu, "Method and software for extracting fuzzy classification rules by subtractive clustering," *IEEE Control Systems Magazine, 1996, vol. pp. 461-465*, 1996.
[14] ——, *Fuzzy Information Engineering: A Guided Tour of Applications*. John Wiley&Sons, 1997, ch. 9, Extracting Fuzzy Rules from Data for Function Approximation and Pattern Classification.
[15] J. C. Bezdek, "Pattern recognition with fuzzy objective function algorithms," *Plenum Press*, 1975.
[16] R. Krishnapuram and C.-P. Freg, "Fitting an unknown number of lines and planes to image data through compatible cluster merging," *Pattern Recognition*, vol. 25(4), pp. 385–400, September 1985.
[17] U. Kaymag and R. Babuska, "Compatible cluster merging for fuzzy modelling," *FUZZ-IEEE/IFES*, vol. 2, pp. 897–904, March 1995.
[18] L. Davis, *Genetic Algorithms and Simulated Annealing*. Morgan Kaufmann, 1987.
[19] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.