Efficient and Interpretable Fuzzy Classifiers from Data with Support Vector Learning

STERGIOS PAPADIMITRIOU[1] KONSTANTINOS TERZIDIS[1] SEFERINA MAVROUDI[2]
LAMBROS SKARLAS[2] SPIRIDON LIKOTHANASIS[2]

1. Dept of Information Management, Technological Educational Institute of Kavala
Kavala, 65404, GREECE

2. Pattern Recognition Laboratory, Department of Computer Engineering and Informatics
School of Engineering, University of Patras, Rion, Patras, 26500, GREECE

Abstract: The maximization of the performance of the most if not all the fuzzy identification techniques is usually expressed in terms of the generalization performance of the derived neuro-fuzzy construction. Support Vector algorithms are adapted for the identification of a Support Vector Fuzzy Inference (SVFI) system that obtains robust generalization performance. However, these SVFI rules usually lack of interpretability. The accurate set of rules can be approximated with a simpler interpretable fuzzy system that can present insight to the more important aspects of the data. The interpretable fuzzy system construction algorithms receive an *a priori* description of a set of fuzzy sets that describe the linguistic aspects of the input variables as they are usually perceived by the human experts. In the case of the interpretable fuzzy sets an adaptive an algorithm for building them automatically is presented here. After the construction of the interpretable fuzzy partitions, the developed algorithms extract from the SVFI rules a small and concise set of interpretable rules. Finally, the Pseudo-Outer Product (POP) fuzzy rule selection orders the interpretable rules by using a Hebbian like evaluation in order to present the designer with the most capable rules.

Key-Words: Interpretable Fuzzy Rules, Rule Mining, Support Vector Machines, Kernel Classifiers

1 Introduction

Support Vector Machine (SVM) are based on Vapnik's Statistical Learning Theory [1] and it has recently received a lot of attention by researchers in many scientific fields. The basic reason is that it can be successfully used in many complex problems [2]. The SVM is an approximate implementation of the *structural risk minimization* inductive principle that aims at minimizing an upper bound on the generalization error of a model, rather than the usual minimization of the mean-square error over the training set. More recently, a new approach to the fuzzy identification problem, based on the SVM, has been developed [9, 10]. This approach offers a robust framework for fuzzy systems that are able to generalize effectively [3, 4]. In this work we make use of algorithms of [9] for the construction of a Support Vector Fuzzy Inference (SVFI) system. The number of SVFI rules is then controlled by the extracted support vectors. The kernel on which the non linear transformation is achieved is that of the Radial Basis Function. That is known as a classifier of type SVM-RBF. Comparisons with clustering based methods for fuzzy rule construction [12, 11, 16, 14, 15, 7, 8] demonstrate that the SVM method has the potentiality of extracting fewer rules that are usually more suitable for a better generalization [9, 10]. The Ockam's razor rule applies here.

Although the support vector approach has the potential of generating effective fuzzy classifiers, these clas-

sifiers are derived by a simple recasting of the SVM inference in fuzzy rule terms. The resulting fuzzy system implements accurately the SVM inference and thus it is generally quite effective. The interpretation of its rules by the human experts is not straightforward, since the rules are defined in terms of the proximity to support vectors, a concept that usually does not provide an intuitive information to the human expert.

The paper presents a simple but effective set of algorithms for the construction of an approximate interpretable fuzzy system from the accurate SVM-based one. This approximate system utilizes interpretable application specific fuzzy sets defined by the domain experts. However, in comparison with the SVFI, it exhibits some loss of classification accuracy. This is not a significant drawback because the SVFI can be used concurrently for retaining the classification efficiency. These advantages can be summarized to:

1. The rules are expressed with domain specific fuzzy sets and thus they can provide direct intuition to the human expert.

2. It has generally much smaller number of rules from its "base" SVFI system and each such rule involves usually a *small subset* of the input features. To the contrary at the SVFI rules every rule consists of the conjunction of clauses, with one clause of the form *CloseTo* for *every* feature dimension.

3. The approximation accuracy and the complexity of the interpretable set of rules can be traded off dynamically by adjusting a set of thresholds.

The paper proceeds as follows:

Section 2 deals with the automatic construction of interpretable fuzzy sets along dimensions for which the human experts cannot predefine them easily. Section 3 reviews the Support Vector Fuzzy Inference system and the corresponding algorithms for fuzzy rule construction from the trained SVMs. This presentation is based on the work of [9]. Section 4 concerns the main contribution of the paper, i.e. the derivation of the interpretable fuzzy rules from the Support Vector Fuzzy Inference rules. The results section (i.e. Section 6) presents applications of the techniques, which deal with both synthetically generated data and real data sets, and evaluates the performance of the extracted rules. Finally, section 7 concludes the work of the paper.

2 Features with unspecified a priori interpretable fuzzy sets

The presented approach requires the a priori specification of interpretable fuzzy partitions for every feature. However, frequently, interpretable fuzzy sets for some features cannot be prespecified by the human experts. At this case the presented algorithm at this section can be used to derive automatically a fuzzy partition that owns interpretability properties.

The interpretable fuzzy set partition construction algorithm for the feature f proceeds by hierarchically merging candidate fuzzy partitions according to the following algorithm:

1. *Initialization* The initial interpretable partition is defined by the obtained Support Vectors (SVs) and consists of the coordinates of the l SVs along the feature dimension f, i.e. l is the initial number of interpretable fuzzy sets.
   Set $m = l$ // m denotes the current number of interpretable fuzzy sets

2. *Hierarchical Merging to obtain K, $2 \leq K \leq l$ fuzzy sets*
   while $m > K$ **do** // merge properly interpretable fuzzy sets until K remain
   * evaluate the criterion D_m for merging fuzzy sets
   * $s = 1$
   * merge fuzzy sets s and $s+1$, modifying neighboring fuzzy sets
   while $s < m - 1$ **do**
   * evaluate D^s_{m-1}
   * restore base partition
   * $s = s + 1$
   end while
   * select and store the partition FP^{m-1} for which: $D_{m-1} = argmax(D^s_{m-1})$
   * base partition = FP^{m-1};
   * $m = m - 1$
   end while

The key point for the effective generation of interpretable fuzzy partitioning is the design of the proper distance metric D_m that will best separate the m fuzzy sets. A clever design for D_m based on the concepts of

external and internal distances is proposed in [17].

3 Support Vector Fuzzy Inference (SVFI) learning

This section reviews the framework for Support Vector Fuzzy Inference (SVFI) proposed in [9]. This method provides a solid foundation for obtaining generalization and over-fitting prevention ability. The main contribution of the current work is the construction of a reduced set of interpretable rules from the SVFI that pinpoint many interesting aspects of the data set in easily conceivable representation for the human expert. The corresponding interpretable rule set does not claim to maximize the classification accuracy. Its purpose is to discover important aspects of the dataset and perhaps to help in elaborating domain specific knowledge. The rules discussed at the current section are expressed in terms of the Support Vectors and implement accurately the SVM inference. Thus, the presented methodology by utilizing concurrently two sets of rules offers both interpretability and accuracy.

We proceed by describing the rules that correspond directly to the SV inference, referred to as the *SV-Inference* rules. The SV-Inference rules are extracted by utilizing the algorithms of Chen & Wang [9]. Since the presented interpretable rule system is constructed "on-top" of the SVFI system we present a self-contained description of these algorithms in order to preserve the continuity of the presentation and to clarify the close coupling of the two systems (i.e. the SVFI and the interpretable rule systems).

The general form of the Support Vector Fuzzy Inference (SVFI) rules is:

Rule k: **if** P_1^k **and** P_2^k **and** ... P_N^k then c_k

where $P_i^k, i = 1, \ldots, N$ are fuzzy clauses, of the form

x_i *is CloseToSV(k, i),*

that test the membership of the ith "coordinate" x_i of the input vector $\mathbf{x} = [x_1, \ldots, x_N]$ at the ith fuzzy set of the kth SV, *CloseToSV(k, i)*. The later sets *CloseToSV(k, i)* fuzzify the numerical distance of the x_i input coordinate to the x_i^k coordinate of the kth support vector. A Gaussian function of the form: $\mu_i^k(x_i) = \exp(-\frac{1}{2}\left(\frac{x_i^k - x_i}{\sigma_k}\right)^2)$, computes the membership by quantifying the proximity of the input value component x_i to the value x_i^k of the ith component of the support vector SV_k. Also, the parameters c_k are real constants, i.e. $c_k \in \Re$. We choose product as the fuzzy conjunction operator, addition for fuzzy rule aggregation and Center Of Area (COA) defuzzification. The resulting model becomes a special case of the Takagi-Sugeno (TS) fuzzy model.

The input-output mapping $\mathbf{F}'$ that the SVFI model performs and the decision function for classification problems $\mathbf{F}(\mathbf{x})$ can be expressed as

$$\mathbf{F}'(\mathbf{x}) = \frac{\sum_{k=1}^{M} c_k \prod_{i=1}^{N} \mu_i^k(x_i)}{\sum_{k=1}^{M} \prod_{i=1}^{N} \mu_i^k(x_i)} = \sum_{k=1}^{M} c_k R_k(\mathbf{x}) \quad (1)$$

$$\mathbf{F}(\mathbf{x}) = sgn\{\mathbf{F}'(\mathbf{x})\} \quad (2)$$

where $\mathbf{x} = [x_1, \ldots, x_N]^T \in \Re^N$ is the input and M is the number of rules. Also, N is the number of conjunctive clauses of the kth rule which is equal to the dimensionality of the input vector $\mathbf{x}$ and $\mu_i^k(x_i)$ computes the membership of the input variable x_i in the fuzzy set *CloseToSV(k, i)*. The term contributed by rule k to the numerator of Equation 1 is $c_k \cdot \prod_{i=1}^{N} \mu_k^i(x_i)$. The constant c_k is derived from the Lagrange multipliers α_i and the training patterns labels y_i as: $y_k = \alpha_k \cdot y_k$. The *relative strength*, $R_k(\mathbf{x})$, of the kth rule expresses how much this rule is involved at the decision for input vector $\mathbf{x}$ and is:

$$R_k(\mathbf{x}) = \frac{\prod_{i=1}^{N} \mu_i^k(x_i)}{\sum_{k=1}^{M}(\prod_{i=1}^{N} \mu_i^k(x_i))} = \frac{K^{'}(\mathbf{x}, \mathbf{z}_k)}{A} = \frac{\exp\left(\frac{-1}{2}\| \frac{\mathbf{x}-\mathbf{z}_k}{\sigma_k} \|^2\right)}{A} \quad (3)$$

where $\mathbf{z}_k = [z_k^1, z_k^2, \ldots, z_k^N]^T \in R^N$ controls the location parameters (i.e. the Gaussian centers) of $\mu_i^k, k = 1, \ldots, M,\ i = 1, \ldots, N$. The denominator $\sum_{k=1}^{M}(\prod_{i=1}^{N} \mu_i^k(x_i))$ attains a constant value denoted by A. Since the denominator is constant it does not affect the classification decision and thus we consider equivalent problems, already scaled properly. Therefore, at the expansion of equation 4 (presented below) we consider only the numerator $K'(\mathbf{x}, \mathbf{z}_k)$.

The decision rule of the output of the fuzzy system of equation 1 can be expressed in terms of kernel functions as:

$$F(\mathbf{x}) = sgn\{\sum_{k=1}^{M} c_k K'(\mathbf{x}, \mathbf{z}_k)\} \quad (4)$$

The kernel K' is a translation invariant kernel defined as $K'(\mathbf{x}, \mathbf{z}_k) = \prod_{i=1}^{N} \mu_i^k(x_i - z_i^k)$ and each μ_i^k membership function is of the familiar one-dimensional Gaussian type. In order to implement $K'(\mathbf{x}, \mathbf{z}_k)$ we use scaled and shifted Gaussians, therefore $K'(\mathbf{x}, \mathbf{z}_k) = K'(\|\mathbf{x} - \mathbf{z}_k\|)$, where the location, of the Gaussians are specified with the location vector $\mathbf{z}_k$.

The SV learning algorithm constructs a fuzzy system with N inputs and M number of rules (one rule for every SV). The number of rules M is derived after the solution to the SVM quadratic programming problem.

The M fuzzy rules can be parameterized with a set of location parameters $\{\mathbf{z}_1, \ldots, \mathbf{z}_M\} \in \Re^N$ for the Gaussian centers that determine the membership functions of the if-part fuzzy rules, and a set of real numbers $(\{c_0, \ldots, c_N\} \in \Re)$ for the constants of the then-part fuzzy rules.

We implemented an approach similar to one presented in [9] for the extraction of Support Vector Fuzzy Inference rules. In order to maintain the compactness of the presentation we formulate below the algorithmic format of the Support Vector Fuzzy Inference (SVFI) learning based on the original work of [9].

Algorithm for SVFI fuzzy classifier identification

1. Construct a classification SVM from the training data to get a decision boundary in the feature space $\mathcal{F}$ of the form $f(\mathbf{x}) = sgn(\sum_{i \in S} y_i \cdot \alpha_i \cdot K(\mathbf{x}, \mathbf{x}_i) + b_0)$ where S is the set of obtained support vectors. Also K is the Gaussian Mercer kernel. This kernel defines implicitly a nonlinear mapping Φ from the input space X to a kernel induced feature space $\mathcal{F}$.

   Assign a suitable value to the regularization parameter C, and solve the corresponding quadratic program to obtain the Lagrange multipliers α_i and a suitable value for the constant bias term b_0. Effective algorithms for training SVMs [13] can be readily utilized at this stage.

2. Extraction of fuzzy rules from the SVM decision rule:
   $r \longleftarrow 0$ // r indexes the rule under construction
   for $i = 1$ **to** l // all training samples l
   if $\alpha_i > 0$ **then** // training samples i with Lagrange multipliers $\alpha_i > 0$ are support vectors
   $r \longleftarrow r+1$ // one more rule corresponding to the current SV will be constructed
   $\mathbf{z}_r \longleftarrow \mathbf{x}_i$ // the location parameter $\mathbf{z}_r$ for rule's r membership functions is the support vector x_i
   $c_r \longleftarrow y_i \alpha_i$ // the value of the singleton type output fuzzy set for rule r
   We denote by $\mathbf{x} = [x_1, x_2, \ldots, x_N]$ the feature values of an input vector $\mathbf{x}$ and by $\mathbf{z}_r = [z_{r_1}, z_{r_2}, \ldots, z_{r_N}]$ the corresponding feature values of the support vector r. The constructed fuzzy rule takes the form:
   if *CloseToSV*(x_1, z_{r_1}) **and** *CloseToSV*(x_2, z_{r_2}) **and** ... **and** *CloseToSV*(x_N, z_{r_N}) **then** y **is** c_r
   Compute the weight of rule r as: $w_r \longleftarrow \alpha_i$ // the magnitude of the Lagrange multiplier signifies the importance of the corresponding rule

   end if
   end for

4 Interpretable rules

Linguistic rule extraction is a very important issue within Knowledge-Based Neurocomputing. Support Vector Machines as well as the equivalent SVFI system interpolate relatively easily large sets of data and provide a means for effective generalization.

However, the SVFI approach has the following basic drawbacks:

- The SVFI rules are formulated with fuzzy sets defined in terms of the feature coordinates of the support vectors (i.e. the *CloseToSV()* fuzzy sets). These later sets usually *do not have a particular meaning* to the human expert.
- For problems with large input feature space dimensionality N the obtained rules involve N conjunctive clauses and it is very difficult to comprehend them intuitively.
- When the number of support vectors becomes large the corresponding large SVFI rule base imposes additional interpretability problems.

Therefore, the derivation of interpretable and comprehensible to the human expert fuzzy rules from the SVFI rules is a very important task since it offers the potentiality for a readable and intuitive knowledge representation. The presented framework constructs rules

that are expressed in terms of concepts that the human expert can understand easily. We develop a completely novel and effective framework for the extraction of interpretable rules from the SVFI when the interpretable fuzzy sets for a feature can be prespecified by the human expert.

The presented framework performs an Interpretable Fuzzy Set (IFS) approximation to the SVFI system with one based on *a priori specified interpretable fuzzy sets*. Specifically, for each feature dimension f the domain expert can define a set of interpretable fuzzy sets that are meaningful. For the particular application domain. For example at a medical diagnosis application, for the *ArterialPressure* feature, fuzzy sets such as *VeryLow*, *Low*, *Medium*, *High*, *VeryHigh* can offer direct interpretation and intuition. Clearly, according to the application domain of interest, we have to decide on the fuzzy set types for the interpretable fuzzy sets (e.g. triangle shaped, trapezoids, Gaussian etc.) and on their names (proper names improve the readability of the extracted interpretable rules). At the system we have developed, a graphical Java interface allows the user to define conveniently these characteristics of the fuzzy system.

After the explicit definition of domain specific fuzzy sets the task of generating fuzzy rules that are expressed in terms of these sets from the SVFI system is completely computational and proceeds without the intervention of the human expert.

For a support vector $\mathbf{sv}$ with scalar value sv_f, $sv_f \in \Re$, for its feature dimension f, the degree of membership $\mu_{IFS_{f,i}}(sv_f)$, of sv_f at every Interpretable Fuzzy Set $IFS_{f,i}$ of feature f, is evaluated. Since the emphasis is on obtaining a small set of interpretable and comprehensible rules we keep as a candidate for clause generation involving feature f only the interpretable fuzzy set at which sv_f obtains the maximum membership, denoting it as $IFS_{f,max}$. We consider the case that sv_f is "sufficiently within" the interpretable fuzzy set $IFS_{f,max}$ of feature f, i.e. $\mu_{IFS_{f,max}}(sv_f) > \beta$, where $\beta \in \Re$ is a threshold parameter. At this case, for the $CloseTo(x_f, sv_f)$ fuzzy clause constructed with the SVFI algorithm of Section 3 we create an approximate interpretable fuzzy clause in terms of $IFS_{f,max}$. The threshold parameter β determines the number and the quality of the derived rules. Clearly, with larger thresholds we construct fewer rules but of better quality.

The membership values $\mu_{IFS_{f,max}}(sv_f)$ are used to compute a measure of the accuracy with which the original SVFI rule is approximated. Specifically, for a possible interpretable rule r extracted from the support vector $\mathbf{sv}$ we define a *Support Vector Rule Similarity* $(SVRS_r)$ parameter as:

$$SVRS_r = \prod_{f=1}^{N} \mu_{IFS_{f,max}}(sv_f) \tag{5}$$

N is the dimensionality of $\mathbf{sv}$. Thus, the $SVRS_r$ parameter for an interpretable rule r is defined as a product of the similarities of all the interpretable clauses and the corresponding SVFI clauses (i.e. the parameters $\mu_{IFS_f}(sv_f)$). The product is justified by the conjunctive structure of the rules.

The construction of the "then" part and therefore of the class label of the rules is straightforward and depends on the sign of the c_r parameters that constitute the "then" part of the SVFI rules. The c_r parameters are computed with the SVFI algorithm presented in Section 3. However, a bit more technical is the extraction of information for the strength of each interpretable rule from the SVFI training results. To accomplish this, we detect the minimum and maximum values of the values $c_r = y_r \cdot \alpha_r, c_r \in \Re$. According to the Support Vector Machine theory [5], the range of c_r values depends on many factors, e.g. the specific problem, the particular training set, the RBF-SVM parameters C (complexity regularization parameter) and σ (spreading of Gaussian centers parameter) etc. Although, in absolute terms the c_r values do not have a particular meaning, their relative magnitude indicates the "weight" (or significance) of the corresponding rule. Therefore, an additional rule pruning step can be performed by avoiding to consider those SVFI rules that do not contribute significantly either for the positive or the negative class.

As a particular example, one rule with $c_r = 0.8$ is a "weak" one if the class range is $[-31.7, 35.2]$ since it affects slightly the classification but the same rule is a "strong" in favor for the positive class one if the class range is $[-0.8, 0.9]$. Therefore, in order to obtain effectively the "weight" w_r of each rule r, we detect the minimum and maximum values of the class range (i.e. values c_r) and we normalize this range to $[-1.0, 1.0]$. The normalization unbiases the weight parameter from the range of c_r values. This "weight" parameter corresponds to the strength of the corresponding rule at the SVFI system.

Recapitulating, the weight parameter w_r quantifies the classification strength of the SVFI rule, while the formely described Support Vector Rule Similarity

($SVRS_r$) parameter, the accuracy of its interpretable rule "version". Thus, the multiplication of the weight parameter w_r with the parameter, $SVRS_r$, adjusts the weight of the interpretable rule considering also the accuracy of the interpretable rule in representing the original SVFI rule. We denote the combined quality measure for each interpretable fuzzy rule r as $Significance_r$, i.e.

$$Significance_r = w_r \cdot SVRS_r \quad (6)$$

A useful concept, especially for high dimensional datasets, for the reduction of the syntactic complexity of the interpretable rules is the one of the *default* interpretable fuzzy set. At the frequent case where a variable most often attains the highest memberships to a particular fuzzy set, that fuzzy set can be treated as the default interpretable fuzzy set. Clauses expressed in terms of the default interpretable fuzzy sets are not displayed explicitly at the representation of the interpretable fuzzy rules. For example, since most genes at a gene expression expreriment are not affected significantly by the experiment's condition, the linguistic variable "Unchanged" can be implicitly assumed for all genes not appearing at the clauses of an interpretable rule.

Frequently, we can specify easily interpretable fuzzy sets for many input feature dimensions. For example, at a gene expression analysis experiment with normalized data where $-1(+1)$ is the maximum underexpression (overexpression) and the value 0 (zero) corresponds to absolutely unaffected genes, we can specify a variety of fuzzy sets according to our apriori knowledge. However there can exist features for which interpretable fuzzy sets, cannot be a priori specified. At these cases we can simply ignore the corresponding feature dimensions at the rule extraction. Alternatively, for those features, data-driven interpretable fuzzy rule extraction algorithms like the hierarchical fuzzy partitioning algorithm proposed in [17] can be utilized.

Below we recapitulate the interpretable fuzzy system construction algorithm in pseudocode format. We recall that the main idea is to replace each of the SVFI clauses *CloseToSV*(x_f, z_{r_f}) by *FuzzyLinguisticVariable*(x_f, z_{r_f}) if the feature dimension f of the support vector $\mathbf{z}_r$ (i.e. z_{r_f}) attains a sufficiently high maximum membership $\mu_{F_{f,max}}(z_{r_f})$ at the *FuzzyLinguisticVariable* fuzzy set $F_{f,i}$.

Algorithm: Extraction of interpretable rules from the SVFI rules

// **Notation:**
// $\mathbf{z}_r$, z_{r_f} : the location parameter of the rth support vector
// and the corresponding feature coordinate f of $\mathbf{z}_r$
// x_f: the input value for the f feature
interpretableClauses = {};
ruleSupport = 1.0;
for all the features f of the support vector $\mathbf{z}_r$ **do**
// replace the clause *CloseToSV*(x_f, z_{r_f}) with a possible interpretable clause
for the interpretable fuzzy set $F_{f,max}$ of the fth feature variable for which z_{r_f} obtains the maximum membership
(e.g. for the interpretable fuzzy sets *HighExpression*, *LowExpression* a value 0.9 will attain maximum membership at the *HighExpression* set)
if $\mu_{F_{f,max}}(z_{r_f}) > \beta$ **then**
// β is the formely described threshold parameter
/* the support vector feature value z_{r_f} attains enough membership to the interpretable fuzzy set $F_{f,max}$, thus concatenate the new clause */
if $F_{f,max}$ is not the default fuzzy set **then**
interpretableClauses = *interpretableClauses* **and** (x_f **is** $F_{f,i}$)
(e.g. x_f can be a gene named BRC (i.e. $V_k \equiv BRC$) and the newly added clause can be: *BRC* **is** *HighExpression*)
// compute a measure of how much the new interpretable rule is supported by the SVM inference rule
ruleSupport = *ruleSupport**$\mu_{F_{f,max}}(z_{r_f})$
endif;
else
/* if even one conjunctive clause cannot have a satisfactory approximation with an interpretable fuzzy set (the default set included) the whole Support Vector rule cannot derive an interpretable rule */
interpretableClauses = {};
return *null*
end else;
end for;

if interpretableClauses != null)
/* interpretable clauses exist, construct the "then" part of the potential interpretable rule that will correspond to the support vector. This construction proceeds by first deciding if the possible rule is sufficiently significant by using the relative magnitude of the Lagrange multiplier. For the positive case we derive the "then" part

as *Class = "Positive"* if the corresponding $b_i = \alpha_i \cdot y_i$ is ≥ 0 and *Class = "Negative"* at the opposite case. */

5 Pseudo-Outer Product Evaluation of the interpretable rules

Another approach for data-driven construction of fuzzy rules is based on Hebbian like learning [2] and is referred as the Pseudo Outer Product (POP) rule. This approach evaluates the compatibility of all the possible rules with the training data and keeps the most promising ones. Specifically, for evaluating the compatibility of a candidate rule with a training pattern both the rule's premise and the rule's consequence truth values are evaluated at the specific data of the training pattern. The larger the product of these truth values is, implies that the better the rule in explaining the corresponding training example is. This evaluation is averaged over the training examples, with explicit provision to avoid considering the noisy ones.

However, the pseudo outer-product (POP) rule identification used in the family of pseudo outer product-based fuzzy neural networks (POPFNN) suffered from an exponential increase in the number of identified fuzzy rules and computational complexity arising from high-dimensional data [18]. This decreases the interpretability of the POPFNN in liguistic fuzzy modeling. This is alleviated with the improved approach of [21] that proposes a Rough Set-based Pseudo Outer-Product (RSPOP) algorithm that integrates the sound concept of knowledge reduction from rough set theory with the POP algorithm. This algorithm performs feature selection through the reduction of attributes and also extends the reduction to rules without redundant attributes.

Adopting a different view we confront also effectively the same concerns by using the interpretable SVFI that avoids the exponential increase of the number of rules. Clearly, the number of rules is bounded by the number of SVs (usually is much smaller). The feature selection of the POP approach [23] is performed by adjusting the structure of the fuzzy neural network by deleting invalid feature inputs according to the identified fuzzy rules. In contrast, we design a feature selection that focuses at the features with significant interpretability in terms of the specified interpretable fuzzy sets. However, although the RSPOP approach yields simpler and thus more interpretable fuzzy systems, the intuitive context of the extracted rules remain unclear. At the context of the current work, we utilize also the POP rule in order to evaluate the effectiveness of the extracted interpretable rules.

The objective of the Pseudo-Outer Product (POP) rule evaluation phase is to compute the degree with which each derived interpretable rule is supported by the training set. The POP evaluation phase computes for each training pattern the degree *antecedentRuleFiring* with which the antecedent part of an interpretable rule fires.

Denote by *antecedentRuleFiring* this degree. Subsequently it checks whether the predicted class aggress with the actual class of the training pattern. If so, it adds the computed *antecedentRuleFiring* to the total score over the training set, otherwise it subtracts it.

6 Results

At this section we demonstrate the potentiality of the presented interpretable rule extraction algorithms from the SVFI systems with two examples:

1. The exact discovery of the XOR boolean function from synthetic data derived from a continuous domain XOR like functional.

2. The approximate uncovering of fuzzy rules that implement a simple gene regulation network, from data generated by sampling the original fuzzy rules.

Since the SVFI system implements accurately the RBF-SVM classification decision function, all the results concerning the generalization potential of the RBF-SVM [1, 5] are valid, and thus we do not elaborate on them. Instead we focus on the results obtained from the interpretable fuzzy rule extraction subsystem.

6.1 XOR-Data: Exact discovery of the XOR boolean function

In order to implement a simple test for the fuzzy rule extraction system, an XOR decision system is used as a first example. XOR is the well known logical (i.e. boolean) function that with two logical variables x_1 and x_2 as inputs it produces output $y = 1$ when $(x_1 = 0, x_2 = 1)$ or $(x_1 = 1, x_2 = 0)$ and $y = 0$ when $(x_1 = 1, x_2 = 1)$ or $(x_1 = 0, x_2 = 0)$. Even though the rules are boolean, we desire to extract such boolean logic

from numerical data. Thus, for the XOR learning problem we use numerical data obtained from the function $y = -x_1 \cdot x_2$. Clearly this function evaluates precisely the logical XOR for the numbers -1 and 1 considering them as *false* and *true* respectively. The other values of the output are classified as false or true depending on the sign of y, i.e. *false* for the negative sign and *true* for the positive one. We generated 50 examples by producing uniform random values for x_1 and x_2 at the range $[-1, 1]$, computing the corresponding $y = x_1 \cdot x_2$ and outputing as class label the sign of y (i.e. $sgn\{y\}$).

We have derived a SVFI system with 8 rules, one rule for every one of the 8 support vectors constructed by the RBF-SVM learning algorithm. However these rules do not reveal anything about the continuous XOR-like function that underlies the production of the synthetic data. On the contrary, the derivation of interpretable rules from the SVFI rules clearly uncovers the XOR logical rules.

6.2 Blind discovery of fuzzy rule systems

In order to test the efficiency of the interpretable SV-based rule extraction approach we conducted experiments with synthetic data generated by randomly sampling the operation of known fuzzy rule systems. The objective is to test the efficiency of the algorithms at uncovering the fuzzy rules from which the data were generated, by using only the data samples without any *a-priori* knowledge about the data generating rules.

We use a simple gene regulation network as the generator of controlled synthetic data for training. The small gene regulation example consists of three genes treated as variables that receive continuous values at the range -1 to 1, with -1 meaning totally underexpressed ("Low"), 0 totally unaffected by the experiment and 1 totally overexpressed ("High"). The continuous range of values between these extremes fuzzifies the concept of gene expression, as usually, e.g. a value of 0.8 for one gene signifies larger relative expression level at the particular experiment from a value of 0.7. The *GEnchancer* is an enhancer gene i.e. one that its expression enhances the level of expression of the control gene *GControl*. Similarly, the *GRepressor* gene is a suppressor gene for GControl, i.e. its expression tends to block the expression of the control gene *GControl*.

We generated training data sets by randomly sampling the input variables *GEnhancer* and *GRepressor* (taking about 50 samples). Consequently we conclude at the value of *GControl* variable by evaluating the fuzzy rule system. In order to treat the learning task as an SVM classification problem we discretize the positive cases of the outcome *GControl* to 1 (i.e. overexpression) and the negative one to -1 (i.e. underexpression).

Although the system has not discovered exactly the original rules, the functions of the *Enchancer* gene and the *Repressor* one with respect to the control gene *GControl* are revealed.

7 Conclusions

With this work, an extension of the work of [9] by building interpretable fuzzy-rule systems on top of the SVFI algorithms is proposed in [9].

The algorithm runs in two phases. During the first phase, the SVFI framework of [9] is utilized as a disciplined method for the generation of fuzzy if-then rules from the training data that is capable of achieving remarkable generalization performance. Concerning applications that involve high-dimensional feature spaces, Support Vector Machine (SVM) can work very effectively at a high (or even infinite) dimensional feature space. Since the constructed Support Vector Fuzzy Inference (SVFI) system can mimick with high accuracy the support vector machinery it owns all of its generalization efficiency.

On the other hand SVFI system doesnt exhibit the desired property of interpretability and cognitive meaning to the human experts domain. During the second phase, we extract another set of rules by examining the structure of the SVFI rules. At this phase we take as an *a priori* "bias" an approximate specification of the fuzzy sets that the human experts consider as relevant to the application domain of interest. The derived set of rules at this stage although is not as powerful as the SVFI system, can still offer very useful insight to the structure of the data. The constructed rules tend to be meaningful, since they are stated in terms of the fuzzy sets defined by the domain experts. Concise rules that highlight aspects of the data generation process can also be revealed.

To sum up, the presented interpretable rule extraction algorithms from the SVFI systems reveal a strong potential which can easily be justified by the following two examples:

1. The revealing of fuzzy rules that implement a simple gene regulation network, from data generated by sampling the original fuzzy rules.

2. The discovery of useful and simple rules from a real gene expression dataset concerning cancer tissue classification.

The evaluation of the generalization performance of the interpretable rule systems and the corresponding performance degradation relatively to the SVFI system was presented here. It was shoen that the performance strongly depends on the particular interpretable fuzzy partition defined by the domain expert. Additionally, we evaluated the interpretable rule basis performance parameters, i.e. the coverage and the precision of the rules. The implementation of the algorithms was done in the Java programming language (SUN-JAVA) with the help of the LibSVM library as the base implementation of the Support Vector Learning [13]. Both the constructed accurate fuzzy system, and the interpretable approximations can be analyzed and evaluated with a fuzzy expert system inference engine implemented in Java. Future work continues with the elaboration of the interpretable fuzzy system construction with algorithms that adapt the interpretable membership functions in the spirit of [11, 12]. The code is available upon request from the corresponding author.

Acknowledgment

This work was partially supported from a European Union funded EPEAK II project "Arximidis", code 04-3-001/5, performed at the Technological Educational Institute of Kavalas, Dept. of Information Management, Greece. Post-doc researcher S. Mavroudi was supported by the Greek Sholarship Foundation with a post-doc sholarship.

References

[1] V. N. Vapnik., 1998, Statistical Learning Theory, New York, Wiley 9. V. N. Vapnik, "An Overview of Statistical Learning Theory", IEEE Trans. On Neural Networks, Vol. 10, No 5, 1999, pp. 988-999

[2] Simon Haykin, Neural Networks, MacMillan College Publishing Company, Second Edition, 1999

[3] B. Scholkopf, S. Mika, J. C. Burges, P. Knirsch, K.-R. Muller, G. Ratsch and A. Smola, "Input Space Versus Feature Space in Kernel-Based Methods", IEEE Trans. On Neural Networks, vol. 10, no. 5, 1999.

[4] B. Scholkopf, A. J. Smola, R. C. Williamson, P. L. Bartlett, "New support vector algorithms", Neural Computation:1207-1245, 2000

[5] Bernhard Scholkopf, Alexander J. Smola, "Learning with Kernels: Support Vector Machines, Regularization and Beyond", MIT Press 2002

[6] J. S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system", *IEEE Trans. Syst. Syst. Man Cybern.*, vol. 23, pp. 665-685, May/June 1993

[7] J. A. Dickerson and B. Kosko, "Fuzzy function approximation with ellipsoidal rules", *IEEE Trans. Syst. Man, Cybern.*, vol. 26, pp. 542-560, Aug. 1996

[8] Bart Kosko, *Fuzzy Engineering*, Prentice Hall, 1997

[9] Yixin Chen, James Z. Wang, "Support Vector Learning for Fuzzy Rule-Based Classification Systems", *IEEE Transactions on Fuzzy Systems*, Vol. 11, No. 6, December 2003, p. 716-728

[10] Jung-Hsien Chiang, Pei-Yi Hao, "Support Vector Learning Mechanism for Fuzzy Rule-Based Modeling: A New Approach", Vol. 12, No. 1, February 2004, pp. 1-12

[11] D. Chakraborty and N. R. Pal, "A Neuro-Fuzzy Scheme for Simultaneous Feature Selection and Fuzzy Rule-Based Classification", *IEEE Transactions on Neural Networks*, Vo. 15, No. 1, January 2004, p. 110-123

[12] M. J. del Jesus, F. Holfmann, L. Jun Navascues, L. Sanchez, "Induction of Fuzzy-Rule-Based Classifiers With Evolutionary Boosting Algorithms", *IEEE Trans. on Fuzzy Systems*, Vo. 12, No. 3, June 2004, pp. 296-308

[13] Chang, C.-C., Lin, C.J, "LIBSVM: A library for support vector machines",2001, Available on-line:
http://www.csie.ntu.edu.tw/ cjlin/libsvm

[14] S. Papadimitriou, K. Terzidis, Symbolic Adaptive Neuro-Fuzzy Inference for Data Mining of Heterogenous Data, *Intelligent Data Analysis (IDA) journal*, Volume 7 (4), IOS Press, 2003, 327-346

[15] S. Papadimitriou, K. Terzidis, Growing Kernel-Based Self-Organized Maps trained with Supervised Bias, *Intelligent Data Analysis (IDA) journal*, Volume 8(2), IOS Press, 2004, 111-130

[16] S. Papadimitriou, S.D. Likothanassis, Kernel-Based Self-Organized Maps trained with Supervised Bias for Gene Expression Data Analysis, *Journal of Bioinformatics and Computational Biology (JBCB)*, Imperial College Press, Vol. 1, No. 4 (2004) 647-680

[17] S. Guillaume, B. Charnomordic, "Generating an Interpretable Family of Fuzzy Partitions From Data", *IEEE Trans. Fuzzy Systems*, Vo 12, No 3, June 2004, p. 324-335

[18] Ang, K. K., Quek, C. & Pasquier, M., "POPFNN-CRI(S): Pseudo outer product-based fuzzy neural network using the compositional rule of inference and singleton fuzzifier", *IEEE Transactions on Systems, Man and Cybernetics*, Part B, 33 (6), 838-849, 2003

[19] J. S. R. Jang and C. T. Sun, "Functional equivalence between radial basis function networks and fuzzy inference systems", *IEEE Trans. Neural Networks*, vol. 4, pp. 156-159, Feb. 1993

[20] Keller, J.M., Yager, R.R., Tahani, H.C., "Neural network implementation of fuzzy logic", *Fuzzy Sets and Systems*, 45, 1-12, 1992

[21] Kai Keng Ang, Chai Quek, "RSPOP: Rough Set-Based Pseudo Outer-Product Fuzzy Rule Identification Algorithm", *Neural Computation*, Vo 17, No 1, 205-243, 2005

[22] Yaochu Jin, Bernhard Sendhoff, "Extracting Interpretable Rules from RBF Networks", *Neural Processing Letters*, 17 (2), 149-164, 2003

[23] C. Quek, R. W. Zhou, "The POP learning algorithms: reducing work in identifying fuzzy rules", *Neural Networks*, Vo 14, 1431-1445, 2001