Linguistic Rule Extraction from Support Vector Machine Classifiers

Xiuju Fu, Lipo Wang*, GihGuang Hung, and Liping Goh

Institute of High Performance Computing
Science Park 2, Science Park Road, Singapore,117528
School of Electrical and Electronic Engineering
Nanyang Technological University
*Block S2, Nanyang Avenue, Singapore 639798
Email: {fuxj,terence,gohlp}@ihpc.a-star.edu.sg, elpwang@ntu.edu.sg
http://www.ntu.edu.sg/home/elpwang

Abstract

Classification decisions from linguistic rules are more desirable compared to complex mathematical formulas from support vector machine (SVM) classifiers due to the explicit explanation capability of linguistic rules. Linguistic rule extraction has been attracting much attention in explaining knowledge hidden in data. In this paper, we show that the decisions from an SVM classifier can be decoded into linguistic rules based on the information provided by support vectors and decision function. Given a support vector of a certain class, cross points between each line, which is extended from the support vector along each axis, and SVM decision hyper-curve are searched first. A hyper-rectangular rule is derived from these cross points. The hyper-rectangle is tuned by a tuning phase in order to exclude those out-class data points. Finally, redundant rules are merged to produce a compact rule set. Simultaneously, important attributes could be highlighted in the extracted rules. Rule extraction results from our proposed method could follow SVM classifier decisions very well. We compare the rule extraction results from SVM with RBF kernel function and linear kernel function. Experiment results show that rules extracted from SVM with RBF nonlinear kernel function are with better accuracy than rules extracted from SVM with linear kernel function. Comparisons between our method and other rule extraction methods are also carried out on several benchmark data sets. Higher rule accuracy is obtained in our method with fewer number of premises in each rule.

keywords: **data mining, linguistic rules, classification, support vector machines**

1 Introduction

Rule extraction [1][3] [16][22] can increase perceptibility and help human beings better understand decisions of learning models in **data mining** applications. Rule extraction can

also help refine initial domain knowledge since irrelevant or redundant attributes tend to be absent in extracted rules. In future data collections, labor cost can be reduced by skipping redundant or irrelevant attributes. In addition, active attributes can be shown in rules extracted which facilitate classification decision making. Those attributes which are more active compared to others can be highlighted in linguistic rules. Oppositely, other classification models usually are opaque in identifying active attributes.

Rule extraction techniques are usually based on machine learning methods such as neural networks, support vector machines, genetic algorithms (GAs), statistical methods, rough sets, decision trees and fuzzy logic.

For a data set with tens or hundreds of attributes and thousands of data patterns, it is hard to identify the roles of the attributes in classifying new patterns without any aid from learning models. For example, neural networks can be trained on these training samples to abstract essences and store the learned essential knowledge as parameters in the network. However, though essential knowledge has been captured and embedded in the trained neural network, humans can not tell exactly why a new pattern is classified to a class, which is sometimes referred to as "black-box" characteristics of neural networks. In the medical domain, a disjunctive explanation given as a rule "If medical measurement A is a_1, and medical measurement B is b_1,..., Then conclusion" is preferable to a complex mathematical decision function hidden in neural networks.

Rule extraction from neural networks has been an active research topic in recent years. In early rule extraction work [5], Gallant [5] used trained neural networks to develop an expert-system engine and interpret the knowledge embedded in neural network models by IF-Then rules . More than a decade had passed. The capability of rule extraction [6][20][21] had been shown for delivering comprehensible descriptions on data concepts from complex machine learning models.

GA has been widely used for practical problem solving and for scientific modeling. With the capability in searching for desirable solutions in the problem space, GA has been employed for extracting rules from neural networks. Fukumi and Akamatsu [4] used GA to prune the connections in neural networks before extracting rules. Hruschka and Ebecken [8] proposed clustering genetic algorithm (CGA) to cluster the activation values of the hidden units of a trained neural network. Rules were then extracted based on the results from CGA. Ishibuchi et al [9]-[10] used GA to obtain concise rules by selecting important members from the rules extracted from a neural network.

Decision trees are often combined together with neural networks in both pedagogical and decompositional rule extraction approaches [17][18]. In the decompositional approach proposed in [17], neural networks are first trained to extract the essential relationship between the input and output. The relationship is thus embedded in interconnected weights and hidden neurons of trained neural networks. Then decision trees are applied to decompose the relationship between inputs and hidden neurons, as well as the relationship between hidden neurons and outputs. The results from decision trees are combined together to deliver rules.

In recent years, support vector machine (SVM) [2][11] has attracted lots of interest for its capability in solving classification and regression problems. Successful applications of SVM have been reported in various areas, including but not limited to areas in communication, time series prediction, and bioinformatics. In many applications, it is desirable to know not only the classification decisions but also what leads to the decisions. However, SVMs offer little insight into the reasons why SVM has made its final results. It is a challenging task to develop a rule extraction algorithm in order to reveal knowledge embedded in trained SVMs and represent the classification decisions based on SVM classification results by linguistic rules.

In this paper, we propose a rule extraction algorithm RulExSVM (rule extraction from support vector machines) for revealing the relationships between attributes and class labels through linguistic rules. The extracted rules are in IF-THEN forms with hyper-rectangular boundaries. Rules are generated directly based on information of support vectors. Given a support vector of a certain class, cross points between lines, along each axis, extended from the support vector and SVM decision hyper-curves are found. A hyperrectangular rule is derived from these cross-points. Out-class data points which do not have the same class label with the support vector are detected. The hyper-rectangle is tuned by a tuning phase in order to exclude those out-class data points. Finally rules are merged to obtain a more compact rule set.

In this paper, SVM is briefly introduced in Section 2. The rule extraction algorithm RulExSVM is described in Section 3. In Section 4, two examples are presented for illustrating the rule extraction algorithm. There are more experimental results shown in Section 5. Finally, this paper is concluded in Section 6.

2 Support Vector Machine Classifiers

Given a two-class data set with a set of pattern $\{X_i, y_i\}_{i=1}^{N}$. Where $X_i = \{x_{i1}, x_{i2}, ..., x_{in}\} \in R^n$, $y_i \in \{1,-1\}$, and N is the number of patterns. The support vector machine classifier can be considered as a quadratic cost function for solving the following optimization problem:
minimize:

$$\sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j K(X_i, X_j) \quad (1)$$

subject to:

$$0 \le \alpha_i \le C, \quad (2)$$

$$\sum_i \alpha_i y_i = 0. \quad (3)$$

$K(X_i, X_j)$ is the kernel function. $K(X_i, X_j) = \phi(X_i \cdot X_j)$, $\phi(X_i)$ maps X_i into a higher dimensional space. C is called the regularization constant. Each data point X_i is with a α_i. The data points with $0 < \alpha_i \le C$ are support vectors, such as the data points

{A,B,C,D,E, F, G, I, J} shown in Fig. 1 (SVM kernel function is nonlinear). Support vector i with $\alpha_i = C$ falls into the region between two separating hyper-curves. See points I and J in Fig. 1.
Based on the solution of the above optimization problem [23], a decision function is determined using the α_i so obtained:

$$f(X) = \sum_{i=1}^{N_s} \alpha_i y_i K(S_i, X) + b \qquad (4)$$

S_i represents the ith support vector and N_s is the number of support vectors.

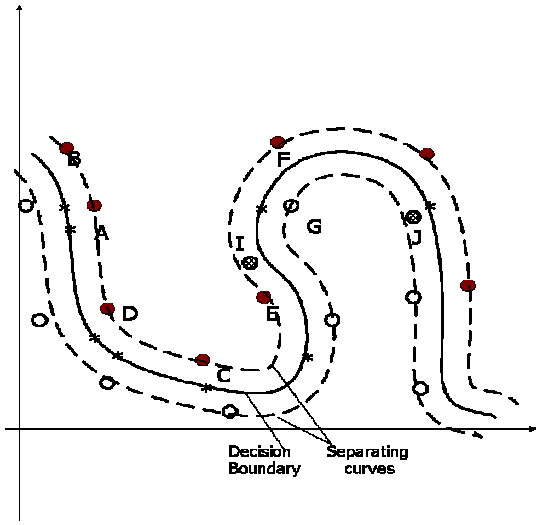

Figure 1: separating hyper-curves, support vectors, and the decision function (© 2005 IEEE) We thank the IEEE for allowing the reproduction of this figure, first appeared in [3].)

3 Rule Extraction

In this section, we describe the rule extraction procedure for a two-class data set. Based on the trained SVM classifier, support vectors of class 1 are used for generating rules for class 1. The rule extraction method described in the later section can be easily extended to multi-class problems. In the rule extraction algorithm, the current class processed is referred to class 1. All the other classes are referred to class 2.

Let us consider an example in the two-dimensional space. SVM classifier is with RBF nonlinear kernel function. In Fig. 2, black points are the support vectors of class 1, and white points are the support vectors of class 2. For each axis, a line, parallel to the axis, starting from a support vector of class 1 is extended in two directions. The cross points between the line and the "decision boundary" can be obtained. The decision boundary refers to the line with $f(X) = 0$. Take for an example, for support vectors A and C, cross points between the extended lines and the decision boundary are shown in Fig. 2. Based on these cross points, the initial boundaries of the hyperrectangular rules can be obtained

and shown as rectangles with dash lines in Fig. 3 (a) and (b) for support vector A and C respectively.

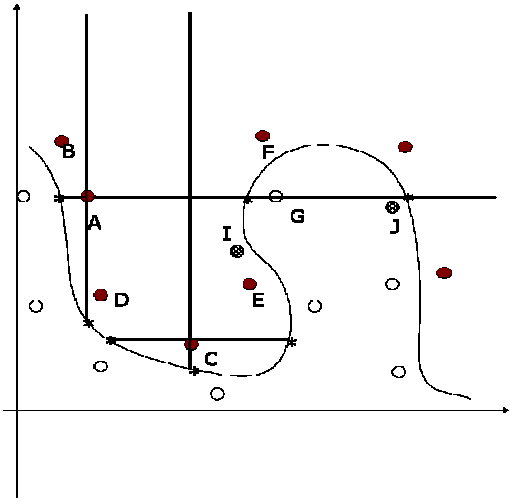

Figure 2: cross points ((© 2005 IEEE) We thank the IEEE for allowing the reproduction of this figure, first appeared in [3].)

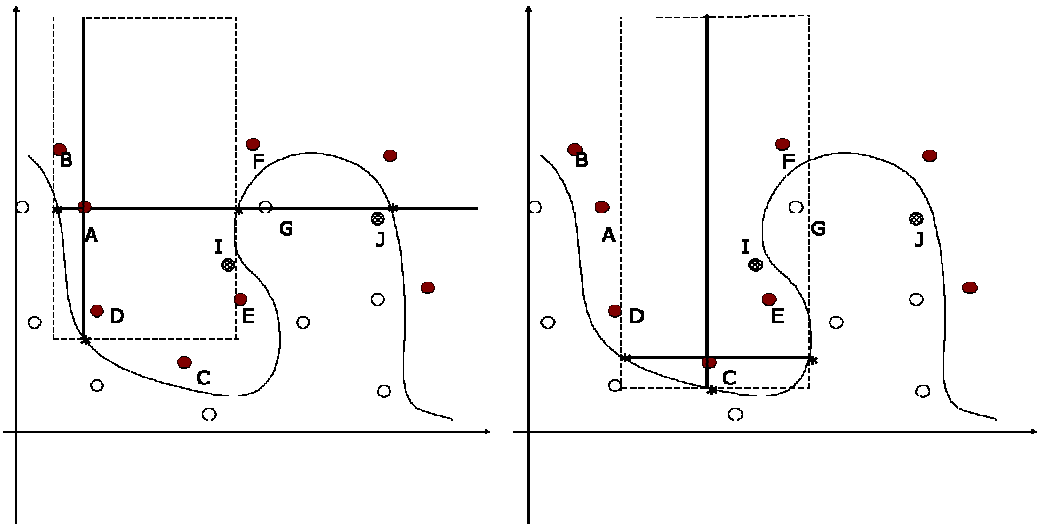

Figure 3: (a) Initial rule generated based on support vector A, and (b) Initial rule generated based on support vector C. ((© 2005 IEEE) We thank the IEEE for allowing the reproduction of this figure, first appeared in [3].)

The rule extraction algorithm from SVM, RulExSVM, consists of three phases, initial, tuning and pruning phases. In the initial phase, given a support vector of class 1, a rule

with hyper-rectangular boundary is generated based on the information provided by the support vector and the decision boundary. In the tuning phase, the initial rule is tuned towards the direction improving the rule accuracy for classifying data. The two phases are stated as below. In the following description of our rule extraction method, SVM classifiers are assumed with RBF kernel functions. The rule extraction procedure based on SVM classifiers with linear kernel functions could be easily figured out accordingly.

3.1 The Initial Phase for Generating Rules

In this section, how to calculate initial hyper-rectangular rules for a two-class data set is stated in detail. The following notation is used. A_i is the support vector set of class i. $i = \{1,2\}$. N_i is the number of support vectors of class i. $N_s = N_1 + N_2$. $S_m = \{s_{m1}, s_{m2}, ...s_{mn}\}$ is the m th support vector of class 1. $X = \{x_1, x_2, ..., x_n\}$ is a pattern of data. Note that all attributes of data points are normalized between [0,1].

The rule with hyper-rectangular boundary derived from a support vector S_m of class 1 can be represented by points included in the rule region:

$$\{X : s_{ml} + \overline{\lambda}_l \geq \overline{x}_l \geq s_{ml} + \underline{\lambda}_l, l = 1,...,n\} \quad (5)$$

subject to $1 \geq \overline{\lambda}_l \geq -1$ and $1 \geq \underline{\lambda}_l \geq -1$.

Let:

$$L_l = s_{ml} + \underline{\lambda}_l \quad (6)$$

and

$$H_l = s_{ml} + \overline{\lambda}_l \quad (7)$$

here H_l and L_l give the upper and the lower limits of the hyper-rectangular rule along the l th dimension. Based on the decision function $f(X)$, L_l and H_l are determined in the following procedure.

Given the solution of a SVM determined by eq. 4 and the corresponding axis l, the rule deriving from support vector S_m can be generated as follows:

1. set $l = 1$, l refers to dimension l
2. Let $X(\lambda) = S_m + \lambda e_l$. Where e_l is a unit vector with all zero elements except the l th element. Define $\tilde{f}(\lambda) := f(X(\lambda))$ as given by eq. 4. Find all possible roots of $\tilde{f}(\lambda)$ with $1 \geq \lambda \geq -1$ by the Newton's method. Let $V_l = \{\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_{kl}\}$ be the set of theses k_l roots of $\tilde{f}(\lambda)$.
3. Determine H_l and L_l. There are several cases to be considered:

(a) $k_l = 0$. This means that there is no cross point between the line extended from S_m along dimension l and the decision boundary.

Hence $H_l = 1$ and $L_l = 0$

(b) $k_l = 1$:

If $\tilde{\lambda}_1 < 0$, $L_l = \left(s_{ml} + \tilde{\lambda}_1\right)$, and $H_l = 1$, else $L_l = 0$, and $H_l = \left(s_{ml} + \tilde{\lambda}_1\right)$

(c) $k_l = 2$. Assume $\tilde{\lambda}_1 < \tilde{\lambda}_2$:

if $\tilde{\lambda}_1 > 0$, then $L_l = 0$, and $H_l = \left(s_{ml} + \tilde{\lambda}_1\right)$

if $\tilde{\lambda}_1 < 0$ and $\tilde{\lambda}_2 > 0$, then $L_l = \left(s_{ml} + \tilde{\lambda}_1\right)$, and $H_l = \left(s_{ml} + \tilde{\lambda}_2\right)$

if $\tilde{\lambda}_2 < 0$, then $L_l = \left(s_{ml} + \tilde{\lambda}_2\right)$, and $H_l = 1$

(d) $k_l > 2$. Assume $\tilde{\lambda}_1 < \tilde{\lambda}_2 < ... < \tilde{\lambda}_{kl}$:

if $\tilde{\lambda}_1 > 0$, then $L_l = 0$, and $H_l = \left(s_{ml} + \tilde{\lambda}_1\right)$

if $\tilde{\lambda}_1 < 0$ and $\tilde{\lambda}_{kl} > 0$, then there exists an index j with $\tilde{\lambda}_j < 0 < \tilde{\lambda}_{j+1}$. Hence, $L_l = \left(s_{ml} + \tilde{\lambda}_j\right)$, and $H_l = \left(s_{ml} + \tilde{\lambda}_{j+1}\right)$

if $\tilde{\lambda}_{kl} < 0$, then $L_l = \left(s_{ml} + \tilde{\lambda}_k\right)$, and $H_l = 1$

4. $l = l + 1$, if $l \le n$, go to Step 2, else end

3.2 The Tuning Phase for Rules

The tuning phase of the RulExSVM method is implemented after an initial rule is generated. The rules generated in the initial phase rely on support vectors and the decision function. Hence it is possible for data points from the other class to lie within the rule regions. It is therefore necessary to redefine the rule by adjusting H_l and L_l to exclude those data points. There are many choices of l varying from 1 to n to achieve the adjustment. We then wish to adjust so that the volume of the tuned hyper-rectangular rule is largest for the various choices of l. The implementation steps of RulExSVM are as follows:

1. Randomly choose a support vector S_m from A1 and invoke the initial phase with S_m as an input. If A1 is empty, then stop.
2. Search all samples of class 2 which are included in the rule region. Let the collection of all such points be Q with cardinality $|Q|$. If $|Q| > 0$, then randomly choose a sample from Q, and go to Step 3, otherwise, remove S_m from A1 and go to Step 1.
3. Adjust the hyper-rectangular rule so that the sample point is outside of the rule region, and that the remaining volume of the rule region is the largest among the possible n dimensions. Go to Step 2.

3.3 The Pruning Phase for Rules

Rules that classify data patterns from different classes may overlap with each other since rules extracted are independent during initialization and tuning phases. If a rule region is totally overlapped by another, it would be considered redundant. The pruning phase remove such redundant rules from the rule set. The pruning procedure is as follows: (1)

find the data points that fall into each rule region, (2) if the set of points in a rule region is a subset of points covered by another rule, the rule is removed, (3) repeat the pruning to remove all redundant rules.

4 An Illustrative Example

In this section, we present an example applying RulExSVM. The data set is a binary-class data set. For a multiple-class data set with M classes, rule extraction is carried out for M binary-class data sets, i.e., one-against-all policy is employed for extracting rules for each class. When training SVM classifiers and extracting rules, we normalize all the attributes to the interval [0, 1]. In the expression of rules, the attributes will be transformed to their original ranges.

The breast cancer data set [13] obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg, has 9 discrete attributes. The original range of the discrete attributes is the interval [1, 10].

RulExSVM extracts rules based on trained SVM classifiers. The parameters of SVM $\{\sigma, C\}$ are determined using 5-fold cross-validation.

Choose a support vector S_m of class 1 randomly, Newton's method is employed to find the initial rule derived from this support vector. In order to reduce the calculation time of Newton's method when searching the cross points between lines extending from the selected support vector along each axis and the decision boundary of support vector machine, a set of values {0, 0.05, 0.1, 0.15, ..., 0.9, 0.95, 1.0} are taken from the interval [0, 1]. Then calculate the values of $\tilde{f}(\lambda)$ (eq. 4) subject to that λ equals to each of the 21 values. In the results of $\tilde{f}(\lambda)$, we are to find two neighbors $\{\lambda \mid \lambda_1, \lambda_2\}$ whose signs of $\tilde{f}(\lambda)$ are different. Let $\lambda = (\lambda_1 + \lambda_2)/2$. If signs of all of $\tilde{f}(\lambda)$'s are the same, λ equals to the value which corresponds to the smallest $\tilde{f}(\lambda)$. This λ serves as the starting point for Newton's method to find the solutions of $\tilde{f}(\lambda) = 0$.

We obtained 7 rules for Breast cancer data set based on SVM classifier with RBF kernel function. The rule accuracy for classification is 97.51%. These rules describe the Benign case, and the Malign case is the default class. The characteristics of the Malign class are considered as the ones opposite to those presented in the rule set.

In [19], 2.9 rules were generated with accuracy 94.04%. Higher rule accuracy is obtained by our method though the number of rules is higher. It is also observed in our rule set that the rule accuracy is obtained without the contribution of attributes 2, 5, 7. When we use the attributes {1, 3, 4, 6, 8, 9} presented in the rule set as the inputs to SVM, we can obtain the same classification result as that obtained by using the whole original attributes as inputs to SVM. It shows that these 3 attributes are not active in determining class labels. This point is an advantage of rule decisions over SVM decision though the accuracy of black-box SVM classifiers is usually higher than the accuracy of rules.

5 Experimental Results

Data sets Mushroom, Breast cancer, and Wine are used to demonstrate the proposed method. These data sets could be obtained from UCI database [14]. The characteristics of data sets used here are shown in Table 1. Discrete and numerical attributes can be found in data sets.

In Table 2, the number of support vectors in SVM classifiers for separating each class from other classes is shown together with classification accuracy based on trained SVM classifiers. The classification results from SVM classifiers with linear and nonlinear kernel functions are listed together in Table 2 The information of rules extracted based on SVM classifiers with nonlinear kernel functions (RBF kernel functions) are shown in Table 3. Rule results based on SVM classifiers with linear kernel functions are shown in Table 4. The number of premises of each rule is calculated on average. In this table, the fidelity shows that rules extracted match SVM classifiers well.

Table 1: Characteristics of data sets used (num-attri: numeric attributes, dis-attri: discrete attributes)

Data sets	patterns	numerical-attri	Discrete-attri	classes
Mushroom	8124	0	22	2
Wine	178	13	0	3
Breast cancer	683	0	9	2

In the experiments, only support vectors are used for generating initial rules and tuning rules by considering training data points. In Table 5, time (seconds) consumed for training SVM classifiers and extracting rules is presented. The rule extraction program is written in C.

In [19], rules were obtained for Mushroom data set with 2 rules, and 98.12% accuracy. For Mushroom data set, 7 rules with 100% accuracy is obtained by the RulExSVM method.

Table 2: SVM classification results

Data sets	SVM accuracy	
	RBF kernel	Linear kernel
Mushroom	100%	98.67%
Wine	99.3%	97.22%
Breast cancer	97.8%	97.25%

Table 3: Rule extraction results from SVM classifiers with RBF kernel (Ruleacc: rule accuracy, Rule-num: number of rules, Prem/P-rule: Premises/per rule)

Data sets	Rule acc	Rule num	Prem/P rule	Fidelity
Mushroom	100%	7	3.33	100%
Wine	99.3%	6	4.3	99.3%
Breast cancer	97.51%	7	5.3	99.27%

Table 4: Rule extraction results from SVM classifiers with linear kernel(Ruleacc: rule accuracy, Rule-num: number of rules, Prem/P-rule: Premises/per rule)

Data sets	Rule acc	Rule num	Prem/P rule	Fidelity
Mushroom	99.08%	6	7.67	97.75%
Wine	100%	4	7	97.22%
Breast cancer	94.2%	4	8	95.64%

Table 5: SVM training time and time spent for rule extraction (CPU 1.3GHZ)

Data sets	RBF kernel (seconds)		linear kernel (seconds)	
	SVM training	rule extraction	SVM training	rule extraction
Mushroom	3.16	24.04	0.78	12.96
Wine	0.03	0.3	0.03	0.06
Breast cancer	0.07	0.22	0.04	0.15

6 Discussion and Conclusion

Training a learning model is considered as a prior step to discover hidden information from data sets. The proposed rule extraction method RulExSVM is composed of 3 main components: first, based on the trained SVM classifier, initial rules are determined by calculating cross points between support vectors and the decision boundary along each axis; second, rules are tuned based on the criterion that excludes data points of other classes from the rule region and keep the rule region as large as possible; third, the rules which are overlapped completely by other rules will be pruned. In this work, we explore rule extraction from SVM classifiers with linear kernel functions and non-linear RBF kernel functions. The rule extraction procedure reported here could be easily extend to SVM with other types of kernel functions.

An example is presented to illustrate how our proposed rule-extraction algorithm RulExSVM works. Our method could be used to extract rules from data sets with discrete or continuous attributes. It is observed that some attributes might not be present in extracted rules. And the roles of present attributes are highlighted in rules. Rule extraction results are more understandable than the decision from SVM classifiers because attributes which make contribution in classifying data samples are observable. For high-dimensional data, the number of rules extracted might be too many. However, it is worthwhile to extract rules since important attributes might be found in rules. And even though there might be only few of interesting rules in the rule set, it is worthwhile to extract rules since domain experts such as doctors might get valuable information by observing objects covered by those interesting rules. On the other hand, Table 3 and Table 4 shows that the fidelity of our rule extraction results corresponding to SVM classification decisions is high. For evaluating a rule extraction algorithm, high rule accuracy is a criterion, but not the only one. Since rules are extracted based on SVM classifiers, it is expected that rule extraction procedure could explain the black-box decision in a linguistic way and reflect the performance of SVM classifiers well. The fidelity is also an important criterion to evaluate rule extraction algorithms.

Rules extracted by our algorithm have hyper-rectangular decision boundaries, which is desirable due to its explicit perceptibility. Rule extraction results from the proposed method could follow SVM classifier decisions very well. Comparisons between the proposed method and other rule extraction methods show that higher rule accuracy is obtained in our method with fewer number of premises in each rule. We believe that rule extraction from SVM classifiers is useful for putting SVM into more practical data mining applications.

References

[1] Bologna G. & Pellegrini C. (1998). Constraining the MLP power of expression to facilitate symbolic rule extraction. *In Proceedings of IEEE World Congress on Computational Intelligence*, *1,* (pp. 146-151).

[2] Burges C.J.C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery, 2(2)*, (pp. 955-974).

[3] Fu X.J., Ong C.J., Keerthi S.S., Hung' G.G. & Goh L.P. (2004). Extracting the Knowledge Embedded in Support Vector Machines. *In IEEE International Joint Conference on Neural Networks, 1,* (pp. 25-29).

[4] Fukumi M. & Akamatsu N. (1998). Rule extraction from neural networks trained using evolutionary algorithms with deterministic mutation. *In The 1998 IEEE International Joint Conference on Computational Intelligence, 1*, (pp. 686-689).

[5] Gallant S.I. (1998). Connectionist Expert Systems*, In Communications of the ACM, 31*, (pp. 152 – 169).

[6] Hofmann A., Schmitz C., & Sick B.(2003). Rule extraction from neural networks for intrusion detection in computer networks. *In IEEE International Conference on Systems, Man and Cybernetics, 2*, (pp. 1259 – 1265).

[7] Hruschka E.R., & Ebecken N.F.F. (1999). Rule extraction from neural networks: modified RX algorithm, *In International Joint Conference on Neural Networks, 4*, (pp. 2504 -2508).
[8] Hruschka E.R. & Ebecken N.F.F. (2000). Applying a clustering genetic algorithm for extracting rules from a supervised neural network. *In Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. 3*, (pp. 407-412).
[9] Ishibuchi H., & Nii M. (1996). Generating fuzzy if-then rules from trained neural networks: linguistic analysis of neural networks. *In IEEE International Conference on Neural Networks. 2*, (pp. 1133-1138).
[10] Ishibuchi H., Nii M., & Murata T. (1997). Linguistic rule extraction from neural networks and genetic-algorithm-based rule selection. *In Proceedings of International Conference on Neural Networks. 4*, (pp. 2390-2395).
[11] Joachims T. (2000). Estimating the Generalization Performance of a SVM Efficiently. *In Proceedings of the Seventeeth International Conference on Machine Learning (ICML).* Morgan Kaufmann.
[12] McGarry K.J., Wermter S., & MacIntyre J. (1999). Knowledge extraction from radial basis function networks and multilayer perceptrons. *In Proceedings of.International Joint Conference on Neural Networks. 4*, (pp. 2494-2497.
[13] Mangasarian O.L. & Wolberg W.H. (1990). Cancer diagnosis via linear programming. *SIAM News, 23, 5*, (pp 1-18).
[14] Murphy P.M., &. Aha D. W (1994). UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science.
[15] Nunez H., Angulo C. & Catala A. (2002). Rule Extraction from Support Vector Machines. *In European Symposium on Artificial Neural Networks.* Bruges (Belgium), d-side publi., ISBN 2930307-02-1, (pp. 107-112).
[16] Saito K. & Nakano R. (1998). Medical Diagnostic Expert System Based on PDP Model*, In IEEE International Conference on Neural Networks, 1*, (pp. 255-262).
[17] Sato M. & Tsukimoto H.(2001). Rule Extraction from Neural Networks via Decision Tree Induction, *In International Joint Conference on Neural Networks. 3*, pp. 1870-1875.
[18] Setiono R.& Liu H. (1996). Symbolic Representation of Neural Networks. *Computer, 29*, (pp. 71-77).
[19] Setiono R.(2000). Extracting M -of-N rules from trained neural networks. *In IEEE Transactions on Neural Networks. 11, 2*, (pp. 512-519).
[20] Setiono R.,. Leow L.W, & Zurada J.M. (2002). Extraction of Rules From Artificial Neural Networks for Nonlinear Regression. *In IEEE Transactions on Neural Networks, 13, 3*, (pp. 564-577).
[21] Tan S.C. & Lim C.P. (2004). Application of an Adaptive Neural Network with Symbolic Rule Extraction to Fault Detection and Diagnosis in a Power Generation Plant. *In IEEE Transactions on Energy Conversion. 19*, (pp. 369-377).
[22] Tsukimoto H. (2000). Extracting Rules From Trained Neural Networks, *In IEEE Transactions on Neural Networks. 11*, (pp. 377-389).
[23] Vapnik V. (1995). The Nature of Statistical Learning Theory. Springer- Verlag, New York, NY.

Xiuju Fu is a research engineer in Institute of High Performance Computing. She received the BS degree and the MS degree, both from Beijing Institute of Technology (China) in 1995 and 1999, respectively. In 2003, she received her Ph.D degree from Nanyang Technological University in Singapore. She has co-authored one monograph, 3 book chapters, and a good number of papers in conference proceedings and journals. Her research areas include: data mining, neural networks, support vector machine, genetic algorithms, classification, data dimensionality reduction, and linguistic rule extraction.

Lipo Wang is (co-)author of over 60 journal publications, 12 book chapters, and 90 conference presentations. He holds a U.S. patent in neural networks and has authored 2 monographs and edited 16 books. He was keynote/panel speaker for several international conferences. He is Associate Editor for IEEE Transactions on Neural Networks (2002-), IEEE Transactions on Evolutionary Computation (2003-), IEEE Transactions on Knowledge and Data Engineering (2005-). He is/was Editorial Board Member of 6 additional international journals. Dr. Wang is Vice President - Technical Activities, IEEE Computational Intelligence Society (2006-2007) and President, Asia-Pacific Neural Network Assembly 2002/2003.

Gih Guang Hung is a Senior Programme Manager at the Institute of High Performance Computing (Singapore) for the Advanced Computing Programme. He leads various groups in research and development in the areas of grid computing, computational intelligence and visualization. Hung graduated from the University of Illinois at Urbana-Champaign in 1993 with a Ph.D. in Electrical Engineering. He currently holds adjunct Associate Professor position with the School of Computer Engineering at the Nanyang Technological University of Singapore. Hung's research interests include high erformance computing techniques, algorithm performance optimization, grid middleware and remote collaboration technology.

Li Ping Goh graduated in 1991 with a Bachelor degree (Hons) in Statistics from Monash University, Australia and obtained her Master's degree from the National University of Singapore in 1994. She was involved in conducting social research on issues related to the public housing in Singapore between 1992 and 1999. In 2000 she joined the Institute of High Performance Computing as a Senior Research Engineer focus in applying data mining techniques to the industry.