Interval Set Representations of 1-v-r Support Vector Machine Multi-classifiers

Pawan Lingras and Cory Butz

Abstract— **Support vector machines (SVMs) are designed for linearly separating binary classes. Researchers have suggested various approaches, such as the one-versus-rest (1-v-r), one-versus-one (1-v-1) and DAGSVM, for applying SVMs to multi-classification problems. The 1-v-r approach tends to have a large training time, while the 1-v-1 and DAGSVM approaches often store a large number of SVMs. We have recently shown how traditional SVMs can be represented using interval or rough sets. In this paper, we extend the interval set formulation of SVMs to classifications that involve more than two classes that are separated using the 1-v-r approach. Our approach possesses several salient features. The presented work is especially useful for soft margin classifiers. Our approach seeks a balance by reducing the training time while storing fewer rules. Finally, our technique provides a semantic interpretation of the classification process, as opposed to the black-box SVM methods.**

Index Terms—**Support vector machines, classification, rough sets, multiclass.**

I. Introduction

MULTI -class classification using support vector machines (SVMs) is a subject of significant interest in recent literature [1,3,4,7,8]. SVMs extend perceptrons, which are probably the earliest classifiers used by the AI community [6,10]. Perceptron-based classifiers were used to classify objects whose representations were linear separable and are inherently designed for binary classifications. The linear separable condition of perceptrons was a serious hindrance in their applicability. Minsky and Papert [6] discussed several problems that could not be solved with the perceptrons. Vapnik [11] proposed SVMs as an alternative to overcome the linearly separable restriction. SVMs use kernel functions that transform the inputs into higher dimensions. With an appropriate choice of kernel function, it may be possible to transform any classification problem into a linear separable case. Moreover, SVMs attempt to find an optimal hyperplane that will maximize the margin between two classes.

In order to apply SVMs to multi-classification problems, it is necessary to change the problem to multiple binary classification problems. Two of the popular approaches are one-versus-rest (1-v-r) and one-versus-one (1-v-1). The 1-v-r solution involves constructing a binary classifier for each class, which separates objects belonging to that class from those that do not. If we assume that there are N classes, the 1-v-r approach will create N binary classifiers. On the other hand, the 1-v-1 approach involves creating binary classifiers for each pair of classes, thereby creating $\frac{N\times(N-1)}{2}$ classifiers. Platt, *et al.* (2000) proposed use of directed acyclic graphs, called DAGSVMs, to reduce the number of computations when deciding classification of objects during the testing and operational phases.

In this paper, we present an interval-set representation of the application of SVMs for multi-classification. Our approach is based on Rough Set theory, proposed by Pawlak [9], which is useful for the classification of objects when the available information is not adequate to represent classes using precise sets. While it may be possible to transform the classification problem by choosing a kernel function with high dimensionality, such a transformation may not be desirable in practical situations. Thereby, we suggest soft margin classifiers, which allow for erroneous classification in the training set. Our approach seeks a balance by reducing the training time while storing fewer rules. Another advantage of our approach is that rough sets provide an explanation of the classification process using logical rules. Such a logical explanation may be useful to users in their decision making process. On the contrary, perceptrons, multi-layered neural networks and SVMs represent a category of classifiers that can be looked at as black-boxes. This is important, since describing the relationship between the black-box approach of network based systems, such as neural networks and SVMs, with the logical rule-based approaches can lead to semantically enhanced network based classifiers. Finally, the work presented here extends our previous work [5] from binary classification to multi-classification.

P. Lingras is with the Department of Math and Computer Science, Saint Mary's University, Halifax, Nova Scotia, Canada, B3H 3C3. (e-mail: Pawan.Lingras@stmarys.ca).

C. Butz is with the Department of Computer Science, University of Regina Regina, Saskatchewan, Canada, S4S 0A2. (e-mail: butz@cs.uregina.ca).

II. Binary and Multiclass Classification with SVMs

Let **x** be an input vector in the input space *X*. Let y be the output in $Y = \{-1, +1\}$. Let $S = \{ (\mathrm{x}_1, y_1), (\mathrm{x}_2, y_2), \dots, (\mathrm{x}_i, y_i), \dots\}$ be the training set used for supervised classification. Let us define the inner product of two vectors **x** and **w** as: $\langle \mathbf{x}, \mathbf{w} \rangle = \sum_j x_j \times w_j$, where x_j and w_j are components of the vectors **x** and **w**, respectively. If the training set is linear separable, the perceptron learning algorithm will find the vector **w** such that:

$$y \times [\, \langle \mathbf{x}, \mathbf{w} \rangle + b \,] \geq 0 \text{, for all } (\mathbf{x}, y) \in S. \quad (1)$$

SVMs overcome the shortcomings of linear separability in the perceptron approach by using a mapping Φ of the input space to another feature space with higher dimension. Equation (1) for perceptrons is then changed as follows:

$$y \times [\, \langle \Phi(\mathbf{x}), \Phi(\mathbf{w}) \rangle + b \,] \geq 0 \text{, for all } (\mathbf{x}, y) \in S. \quad (2)$$

Usually, a high dimensional transformation is needed in order to obtain reasonable classification [2]. Computational overhead can be reduced by not explicitly mapping the data to feature space, but instead just working out the inner product in that space. In fact, SVMs use a kernel function *K* corresponding to the inner product in the transformed feature space as: $K(\mathbf{x}, \mathbf{w}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{w}) \rangle$. Polynomial kernel is one of the popular kernel functions. Let us derive the polynomial kernel function of degree 2 for two dimensional input space. Let $\mathbf{x} = (x_1, x_2)$ and $\mathbf{w} = (w_1, w_2)$.

$$\begin{aligned} K(\mathbf{x}, \mathbf{w}) &= \langle \mathbf{x}, \mathbf{w} \rangle^2 \\ &= (x_1 w_1 + x_2 w_2)^2 \\ &= (x_1^2 w_1^2 + x_2^2 w_2^2 + 2 x_1 w_1 x_2 w_2) \\ &= \left\langle x_1^2 + x_2^2 + \sqrt{2} x_1 x_2,\ w_1^2 + w_2^2 + \sqrt{2} w_1 w_2 \right\rangle \\ &= \langle \Phi(\mathbf{x}),\ \Phi(\mathbf{w}) \rangle \end{aligned}$$

The dimensionality rises very quickly with the degree of polynomial. For example, Hoffmann [3] reports that for an original input space with 256 dimensions, the transformed space with second degree polynomials was approximately 33,000, and for the third degree polynomials the dimensionality was more than a million, and fourth degree led to a more than billion dimension space. This problem of high dimensionality will be discussed later in the paper.

The original perceptron algorithm was used to find one of the possibly many hyperplanes separating two classes. The choice of the hyperplane was arbitrary. SVMs use the size of margin between two classes to search for an optimal hyperplane. The problem of maximizing the margin can be reduced to an optimization problem [2,11]:

$$\text{Minimize } \langle \mathbf{w}, \mathbf{w} \rangle \text{ such that } y \times [\, \langle \mathbf{x}, \mathbf{w} \rangle + b \,] \geq 0 \text{, for all } (\mathbf{x}, y) \in S. \quad (3)$$

SVMs attempt to find a solution to such an optimization problem.

2.2 Multi-classification with SVMs

The problem of multi-classification, especially for systems like SVMs, does not present an easy solution [7]. It is generally simpler to construct classifier theory and algorithms for two mutually-exclusive classes than it is for *N* mutually-exclusive classes. Platt [8] claimed that constructing N-class SVMs is still an unsolved research problem. The standard method for N-class SVMs [11] is to construct *N* SVMs. The *i*th SVM will be trained with all of the examples in the *i*th class with positive labels, and all other examples with negative labels. Platt *et al.* [7] refer to SVMs trained in this way as *1-v-r* SVMs (short for one versus rest). The final output of the *N* 1-v-r SVMs is the class that corresponds to the SVM with the highest output value. Platt *et al.* list the disadvantages of 1-v-r approach as follows. There is no bound on the generalization error for the 1-v-r SVM, and the training time of the standard method scales linearly with *N*.

Another method for constructing N-class classifiers from SVMs is derived from previous research into combining two-class classifiers. Knerr *et al.* [4] suggested constructing all possible two-class classifiers from a training set of *N* classes, each classifier being trained on only two out of *N* classes. Thus, there would be $\frac{N \times (N-1)}{2}$ classifiers. Platt *et al.* [7] refer to this as *1-v-1* SVMs (short for one-versus-one). Platt *et al.* [7] proposed DAGSVM, which uses directed acyclic graphs to reduce the number of SVMs that need to be used during the testing and operational phase. Chang *et al.* [1] studied one-against-one and DAGSVM. In the training phase, both methods require solving $\frac{N \times (N-1)}{2}$ binary classification problems. In the testing phase, the one-against-one technique conducts $\frac{N \times (N-1)}{2}$ classifications, while the DAGSVM technique employs a directed acyclic graph that has $\frac{N \times (N-1)}{2}$ nodes and *N* leaves, reducing the number of classifications to *N*-1. Both methods are subject to the drawback that, when the number of classes *N* is large, they incur exhaustive amount of training time and produce an extremely large set of support vectors [1].

The 1-v-r approach creates *N* SVMs as opposed to $\frac{N \times (N-1)}{2}$ SVMs created and stored for 1-v-1 and DAGSVM methods. The training of 1-v-1 is computationally less expensive, since only a subset (corresponding to the pair of classes involved) of the training sample is used. This paper

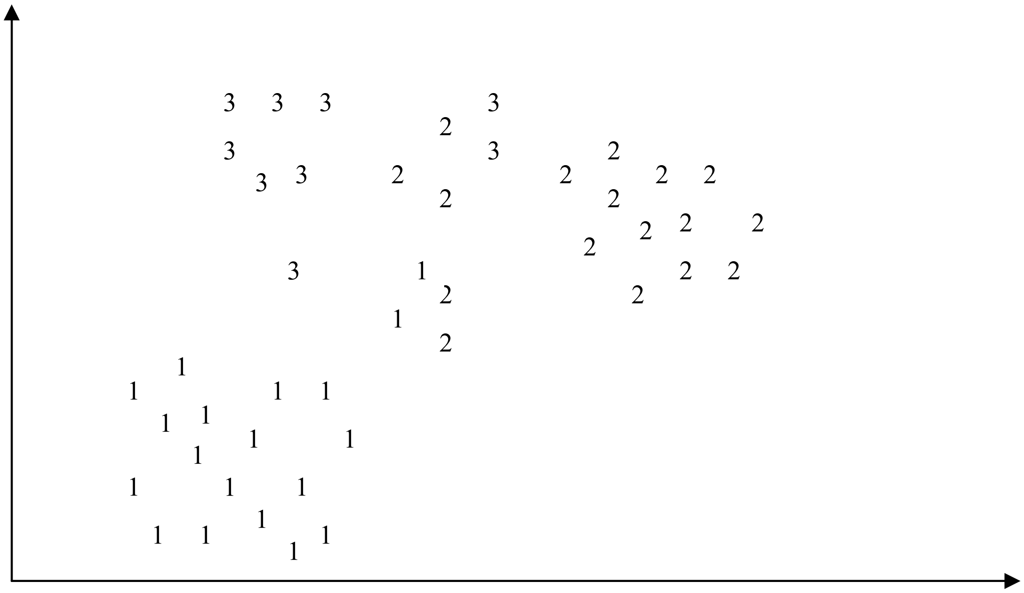

Fig. 1. Three Class Classification Example

describes a rough set based scheme that makes it possible to reduce the training set size, provides a possible semantic description for the multiclassification, and makes the testing and operational phase more streamlined.

III. Rough Sets based on Binary Support Vector Machine Classification

This section describes rough set interpretation of SVM binary classification proposed by Lingras and Butz (2004). A certain familiarity with the theory of rough sets is assumed. We will first consider the ideal scenario, where the transformed feature space is linear separable and the SVM has found the optimal hyperplane by maximizing the margin between the two classes. The optimal hyperplane gives us the best possible dividing line. However, if one chooses to not make an assumption about the classification of objects in the margin, the margin can be designated as the boundary region. This will allow us to create rough sets as follows.

Let us define b_1 as follows: $y \times [\langle \mathbf{x}, \mathbf{w} \rangle + b_1] \geq 0,$ for all $(\mathbf{x}, y) \in S$, and there exists at least one training example $(\mathbf{x}, y) \in S$ such that $y = 1$ and $y \times [\langle \mathbf{x}, \mathbf{w} \rangle + b_1] = 0$. Similarly, b_2 is defined as: $y \times [\langle \mathbf{x}, \mathbf{w} \rangle + b_2] \geq 0,$ for all $(\mathbf{x}, y) \in S$, and there exists at least one training example $(\mathbf{x}, y) \in S$ such that $y =$ -1 and $y \times [\langle \mathbf{x}, \mathbf{w} \rangle + b_2] = 0$. It can be easily seen that b_1 and b_2 correspond to the boundaries of the margin.

The modified SVM classifier can then be defined as follows.

If $\langle \mathbf{x}, \mathbf{w} \rangle + b_1 \geq 0$, classification of **x** is +1. (R1)

If $\langle \mathbf{x}, \mathbf{w} \rangle + b_2 \leq 0$, classification of **x** is -1. (R2)

Otherwise, classification of **x** is uncertain. (R3)

The proposed classifier will allow us to create three equivalence classes, and define a rough set based approximation space. This simple extension of an SVM classifier provides a basis for more practical applications, when SVM transformation does not lead to a linear separable case. Cristianini (2003) list disadvantages of refining feature space to achieve linear separability. Often this will lead to high dimensions, which will increase the computational requirements significantly. Moreover, it is easy to overfit in high dimensional spaces, i.e., regularities could be found in the training set that are accidental, which would not be found again in a test set. The soft margin classifiers (Cristianini, 2003) modify the optimization problem to allow for an error rate. The rough set based rules given by (R1-R3) can still be used by empirically determining the values of b_1 and b_2. For example, b_1 can be chosen in such a way that, for an $(\mathbf{x}, y) \in S$, if $\langle \mathbf{x}, \mathbf{w} \rangle + b_1 \geq 0$, then y must be +1; and, b_2 can be chosen in such a way that, for an $(\mathbf{x}, y) \in S$, if $\langle \mathbf{x}, \mathbf{w} \rangle + b_2 \leq 0$, then y must be -1. Such a choice of b_1 and b_2 would be reasonable, assuming there are no outliers. Otherwise, one can specify that the requirements hold for a significant percentage of training examples. For example, b_1 can be chosen in such a way that, for an $(\mathbf{x}, y) \in S$, if $\langle \mathbf{x}, \mathbf{w} \rangle + b_1 \geq 0$, then in at least 95% of the cases y must be +1.

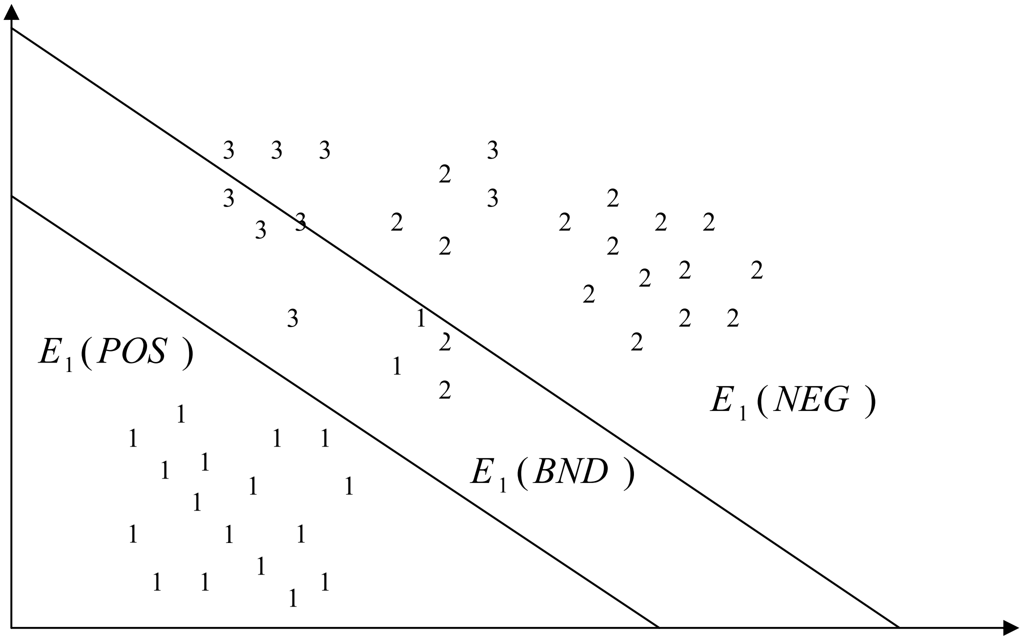

Fig. 2. 1-v-r classification for class 1 represented as rough sets

Similarly, b_2 can be chosen in such a way that, for an $(\mathbf{x}, y) \in S$, if $\langle \mathbf{x}, \mathbf{w} \rangle + b_2 \leq 0$, then in at least 95% of the cases y must be -1.

The extension proposed by Lingras and Butz (2004) can be easily implemented after the soft margin classifier determines the value of **w.** All of the objects in the training sample will be sorted based on the values of $\langle \mathbf{x}, \mathbf{w} \rangle$. The value of b_1 can be found by going down (or up if the positive examples are below the hyperplane) in the list until 95% of the positive examples are found. Similarly, the value of b_2 can be found by going up (or down if the positive examples are below the hyperplane) in the list until 95% of the negative examples are

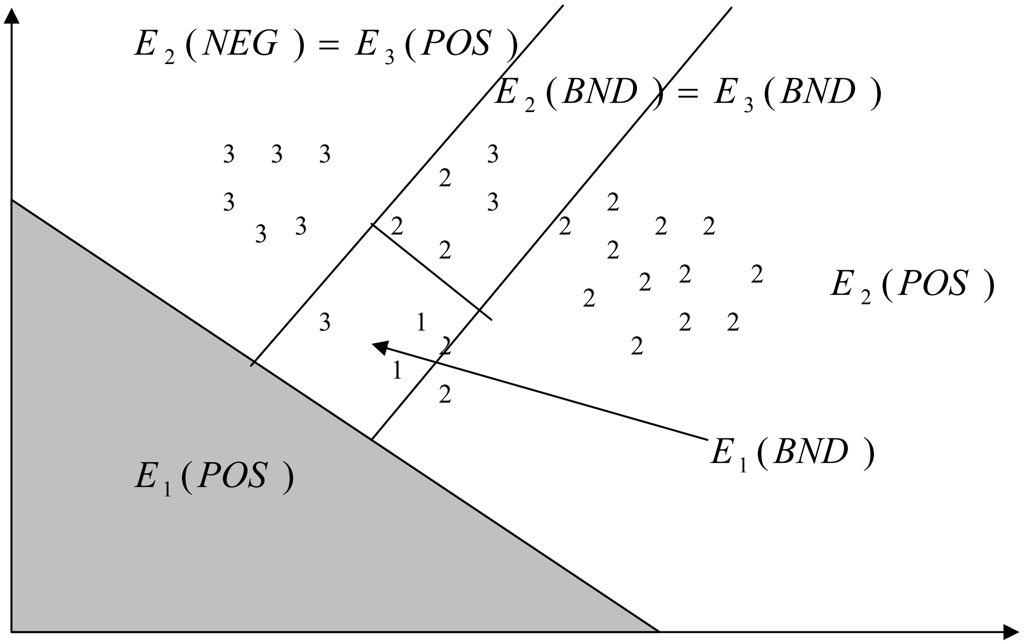

Fig. 3. 1-v-r classification for classes 2 and 3 represented as rough sets

found.

IV. Rough set contributions towards the 1-v-r approach

Figure 1 shows a classification problem with three classes 1, 2 and 3. It is assumed that the objects have already been mapped using the same mapping Φ to transform the problem into a linear separable case. The rough set classification, described by rules R1-R3 in the previous section, can be used to create three equivalence classes for a binary classification problem. Let $E(POS)$ be the set of **x** (or region) that follows rule R1, $E(NEG)$ be the set of **x** that follows rule R2, and $E(BND)$ be the set of **x** that follows rule R3. The lower bound for positive examples will be $E(POS)$ and the upper bound will be $E(POS) \cup E(BND)$. These equivalence classes can be extended to *N* classes as follows.

1. Without loss of generality, let us order the *N* classes such that class *i* contains at least as many objects than class *i*+1, where $1 \le i < N$ in the training sample.
2. For class 1, create three equivalence classes $E_1(POS), E_1(BND), E_1(NEG)$ based on rules R1-R3 and using and the entire training sample and 1-v-r strategy as shown in Figure 2. It is assumed that objects in the region $E_1(POS)$ definitely belong to class 1. $E_1(NEG)$ corresponds to objects that do not belong to class 1, while $E_1(BND)$ may or may not belong to class 1. As $E_1(POS)$ only contains objects belonging to class 1, there is no need to further classify objects in $E_1(POS)$. However, $E_1(BND) \cup E_1(NEG)$ should be further refined, as done in step 3.
3. For each subsequent class *i,* $1 < i < N$, refine $E_{i-1}(BND) \cup E_{i-1}(NEG)$ by creating $E_i(POS), E_i(BND), E_i(NEG)$. Figure 3 shows the resulting classification for class 2. The shaded triangular area in Figure 3 is eliminated from further classification, since it definitely belongs to class 1. 1-v-r classification allows us to identify the objects that definitely belong to class 2. In general case, the process will be further repeated until number of classes is reduced to two. In our example, we stop after applying the 1-v-r classification to class 2, since we have already classified the last two classes.
4. Finally, $E_N(POS) = E_{N-1}(NEG)$ and $E_N(BND) = E_{N-1}(BND)$. Since class 3 is the last class in our example, Figure 3 shows the final classification using the proposed approach. The negative region of class 2 is the same as positive region of class 3, i.e., $E_2(NEG) = E_3(POS)$.
5. As it is possible that some of the boundary regions may overlap with positive regions for subsequent classes, recalculate values of each of the boundary classes as: $E_i(BND) = E_i(BND) - \bigcup_{j=i}^{N} E_j(POS)$. The modified boundary regions are also illustrated in Figure 3.
6. It can now be easily verified that $E_i(POS)$ are mutually exclusive, for $1 \le i \le N$. These will be the equivalence classes of the final approximation space *A* for our *N* classifications. On the other hand, the $E_i(BND)$ might overlap with each other. Therefore, they cannot be equivalences classes of the approximation space *A*. It is possible to divide the collective boundary region into a frame of disjoint sets to define the equivalence classes. However, such an exercise is not necessary for calculating the upper and lower bounds, which can be defined as: $\underline{A}(class_i) = E_i(POS)$ and $\overline{A}(class_i) = E_i(POS) \cup E_i(BND)$.

An important feature of the proposed approach is that the sample size is reduced in each subsequent step. The lower bound of the largest class is eliminated from further classification, followed by the next largest class, and so on. By reducing the size of training set, this elimination process may increase the training performance over the traditional 1-v-r approach. Moreover, only two rules need to be stored for each class, one corresponding to $E_i(POS)$ and another corresponding to $E_i(BND)$. Therefore, a total of 2*N rules are stored for testing and operational phase, as opposed to $\frac{N \times (N-1)}{2}$ SVMs stored by 1-v-1 and DAGSVM approaches. Finally, the rules corresponding to lower and upper bounds may be able to provide a better semantic interpretation of the multi-classification process than the other SVM approaches, which tend to be black-box models.

V. Conclusions

This paper describes an extension of a formulation to represent a classification scheme obtained from a SVM using rough or interval sets. The previous formulation of SVMs using rough sets was applicable to the traditional binary SVM classification. Classification problems, in practice, tend to involve more than two classes. Researchers have proposed several extensions to the traditional binary SVM approach, each with its advantages and disadvantages. The classical 1-v-r approach tends to have large training time, while competing approaches such as 1-v-1 and DAGSVM tend to store a large number of SVMs. The proposed extension of the 1-v-r approach using rough set theory attempts to strike a balance between the two approaches by reducing the training set of the 1-v-r approach (see Figures 2 and 3) and storing fewer rules. The proposed approach may also make it possible to provide rule-based semantic interpretation of the classification process, as opposed to the black-box SVM models.

REFERENCES

[1] F. Chang, C-H. Chou, C-C. Lin, and C-J. Chen, "A Prototype Classification Method and Its Application to Handwritten Character Recognition," *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 2004.

[2] N. Cristianini, "Support Vector and Kernel Methods for Pattern Recognition," Available: http://www.support-vector.net/tutorial.html, 2003.

[3] A. Hoffmann, "VC Learning Theory and Support Vector Machines," Available: http://www.cse.unsw.edu.au/~cs9444/Notes02/Achim-Week11.pdf , 2003

[4] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: A stepwise procedure for building and training a neural network," In *Neurocomputing: Algorithms, Architectures and Applications*, Fogelman-Soulie and Herault, Eds, NATO ASI, Springer, 1990.

[5] P. Lingras and C. Butz, "Interval Set Classifiers using Support Vector Machines," *Proceedings of 2004 conference of the North American Fuzzy Information Processing Society*, Banff, Alberta, June 27-30, 2004, pp. 707-710.

[6] M.L. Minsky, and S.A. Papert, *Perceptrons*, Cambridge, MA: The MIT Press, 1969.

[7] J.C. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin DAG's for multiclass classification," in *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, 2000, pp. 547–553.

[8] J.C. Platt, "Support Vector Machines," Available: http://research.microsoft.com/users/jplatt/svm.html, 2003.

[9] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning about Data,* Kluwer Academic Publishers, 1992.

[10] F. Rosenblatt, "The perceptron: A perceiving and recognizing automaton," *Technical Report* 85-460-1, Project PARA, Cornell Aeronautical Lab, 1957.

[11] V. Vapnik, *Statistical Learning Theory.* New York: Wiley, 1998.