

Building Fuzzy Classifiers with Pairwise Multiclass Support Vector Machines

José M. Benítez

Dept. of Computer Science
and Artificial Intelligence

University of Granada

J.M.Benitez@decsai.ugr.es

Juan L. Castro

Dept. of Computer Science
and Artificial Intelligence

University of Granada

castro@decsai.ugr.es

José M. Puche

Dept. of Computer Science
and Artificial Intelligence

University of Granada

puche@decsai.ugr.es

Abstract

In this paper, we make a proposal to build a classifier with fuzzy outputs with multiclass Support Vector Machines (SVMs). This allows us to widen the applicability of this kind of powerful and soundly founded classifiers. We consider the pairwise multiclass version and investigate different alternatives for the aggregation process. A number of experiments have been carried out to establish the validity and performance of this approach.

Keywords: support vector machines, fuzzy aggregation, multiclass classification

1 Introduction

Support Vector Machines (SVMs) conform a wide set of models based on Statistical Learning Theory. This theory provides sound foundations for models and bounds for their generalisation performances.

Since their introduction in the COLT-92 [1], an increasing number of researchers and practitioners have been investigating and working with them. SVMs have proved successful in a wide area of applications ranging from OCR to Information Retrieval to Image Classification.

SVMs have been designed and applied mainly to classification problems. Actually, crisp classification problems. However, it is well known that many real-world problems are affected by uncertainty. In many situations, the uncertainty can be modelled using Fuzzy Logic and Fuzzy sets.

In this paper, we propose a method to develop a fuzzy classifier with multiclass SVMs. Among the different approaches for multiclass SVMs, we have chosen the pairwise one because it produces the better results and is the most efficient.

Our goal is to develop a model for fuzzy classification, which retains or improve its performance, while widening the applicability of SVMs. We achieve this goal by defining fuzzy membership functions for individual SVMs and aggregating them afterwards with fuzzy logic operators.

A previous proposal in the same line of ours has been made by Tsujinishi et al. [2, 3]. Our method defines different membership functions which renders better performance as empirical results will show.

2 Support Vector Machines

2.1 Description

In this section, we produce a brief introduction to binary Support Vector Machines theory. The interested reader can find a detailed description about Statistical Learning Theory and SVMs in [4, 5] and [6, 7, 8], respectively.

SVMs are classification models which implements the structural risk minimization (SRM) inductive principle[5]. SVMs minimize the risk functional bound by keeping the value of the training error fixed (equal to zero or to some acceptable value) and minimizing the VC confidence.

Let $(x_1, y_1), \dots, (x_l, y_l) \in \mathbb{R}^n \times \{-1, +1\}$ be a set of i.i.d. training samples. SV learning finds out a canonical hyperplane $\{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{w}, \mathbf{x} \rangle - b =$

$0, \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}$ that maximally separates the two classes of training samples. The corresponding classifier is

$$f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle - b). \quad (1)$$

Considering that the training samples may be non-linearly separable, to find the optimal decision boundary one has to solve the following quadratic programming problem:

$$\text{minimize } \Phi(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^l \xi_i \quad (2)$$

under the constraints

$$y_i[\langle \mathbf{w}, \mathbf{x}_i \rangle - b] \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad i = 1, 2, \dots, l, \quad (3)$$

where C is a positive integer constant given for the user at the beginning of the learning process. (2) is solved using Lagrangian optimization theory [6, 7, 8]. The dual optimization problem is as follows

$$W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \quad (4)$$

constrained to

$$\sum_{i=1}^l \alpha_i y_i = 0 \quad \text{and} \quad (5)$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \dots, l. \quad (6)$$

where α_i are the Lagrange multipliers. The corresponding \mathbf{w} dual representation is

$$\mathbf{w} = \sum_{i \in S} y_i \alpha_i \mathbf{x}_i, \quad (7)$$

where S is the set that contains the training samples \mathbf{x}_i with nonzero Lagrange multiplier. This set contains training vectors for which equality is achieved in inequality (3). This fact follows from the Kuhn-Tucker conditions. Training samples with a nonzero Lagrange multiplier are called *support vectors*.

Let S be the index set of support vectors. By substituting (7) in (1), the dual decision function results

$$f(\mathbf{x}) = \text{sgn} \left(\sum_{i \in S} y_i \alpha_i \langle \mathbf{x}_i, \mathbf{x} \rangle - b \right). \quad (8)$$

Heretofore, decision boundary is a canonical hyperplane in \mathbb{R}^n . However, it also possible to use other more complex decision boundaries. To obtain them, it is enough with applying a nonlinear mapping

$$\begin{aligned} \Phi : \mathbb{R}^n &\rightarrow Z \\ \mathbf{x} &\mapsto \Phi(\mathbf{x}), \end{aligned} \quad (9)$$

to map the samples into a new high-dimensional feature space Z . An optimal separating hyperplane is constructed in this space. But there are two problems: *What does Φ function use?* and *how to overcome the high computational cost of such high-dimensional space Z ?* One can observe in (4) and (8) that a training data never appear isolated, but they are always into an inner product. From this fact, it is derived that there is no need to consider the feature space in *explicit form*, but only the inner products between vectors of the feature space Z are necessary. Those can be computed through a *kernel function*, which is defined as follows

$$\begin{aligned} K : \mathbb{R}^n \times \mathbb{R}^n &\rightarrow \mathbb{R} \\ (\mathbf{x}_i, \mathbf{x}_j) &\mapsto K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle. \end{aligned} \quad (10)$$

A function fulfilling Mercer theorem [5, 9] is a kernel functions. Kernels are introduced in SVMs by substituting $K(\mathbf{x}_i, \mathbf{x}_j)$ for $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ in every expression.

Finally, SVMs can be trained with several techniques [10, 11, 12]. Particularly, we have used the Sequential Minimal Optimization (SMO) algorithm [12].

2.2 Pairwise Multiclass SVMs

SVMs were originally designed for binary classification, but they can also be extended to solve multiclass classification. Currently there are two approaches for multiclass SVM. One is based upon the construction and combination of binary classifiers. The other uses all data in one optimization problem [5, 9, 13]. Here, we only consider the first approach.

There are two major approaches to combine binary classifiers: “one-against-all” [14] and “one-against-one”. The last method constructs $k(k -$

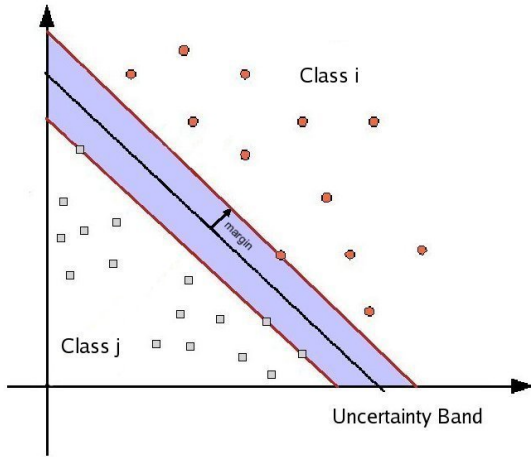


Figure 1: An SVM hyperplane separating classes i and j , with its corresponding band.

1)/2 classifiers where each one is trained on samples from two classes, while the other requires k classifiers and each one is trained on data from one class against the rest of classes. There are several aggregation methods for both approaches.

For the “one-against-one” approach, there are two standard aggregation strategies: one is the voting strategy, the first use of which in SVMs was in [14], and the other is the directed acyclic graph SVM (DAGSVM) [15]. Both of them have their advantages and weaknesses. In particular, both of them are unable to provide a multiple class answer, which is necessary, when there are uncertainty in the outputs. For instance, although formally correct, it is “risky” to make a decision when the computed output is 0.05. In next section, to alleviate these deficiencies we define a procedure to build a classifier with fuzzy outputs based on SVMs.

3 Fuzzy Classifiers based on SVMs

SVMs were first proposed and are mainly used to solve crisp classification problem. However, they can be easily used to build a classifier with fuzzy outputs. Actually, by using different aggregating operators, this can be achieved in different ways. We describe some of them in this section.

Our main goal is to widen the kind of classification problems to which SVM can be applied, by incorporating problems where the output is a

fuzzy set in the class referential. This should be reached while preserving or improving the classifier performance, of course.

To reach the former goal, our basic idea is to express the output of individual SVM in terms of fuzzy membership functions, and then aggregate these outputs with Fuzzy Logic operators. As in the pairwise multiclass approach, for each pair of classes a binary SVM is built. Let $f_{ij}(\mathbf{x})$ be the decision function for the pair of classes i and j . We introduce $(k - 1)$ fuzzy membership functions for every class. These membership functions are built in the direction orthogonal to the optimal separating hyperplane. From a (binary) SVM that classifies class i against j , two membership functions are obtained, namely, $\mu_{ij}(\mathbf{x})$ and $\mu_{ji}(\mathbf{x})$. $\mu_{ij}(\mathbf{x})$ is the degree of membership of \mathbf{x} to the class i according to $f_{ij}(\mathbf{x})$. $\mu_{ji}(\mathbf{x})$ gives the same information with regard to class j .

Without loss of generality, we suppose that $f_{ij}(\mathbf{x})$ is positive when \mathbf{x} is classified as belonging to the class i . Hence, we can define $\mu_{ij}(\mathbf{x})$ and $\mu_{ji}(\mathbf{x})$ as follows:

$$\mu_{ij}(\mathbf{x}) = \begin{cases} 1, & f_{ij}(\mathbf{x}) \geq 1 \\ 0.5 + f_{ij}(\mathbf{x})/2, & -1 < f_{ij}(\mathbf{x}) < 1 \\ 0, & f_{ij}(\mathbf{x}) \leq -1 \end{cases} \quad (11)$$

$$\mu_{ji}(\mathbf{x}) = 1 - \mu_{ij}(\mathbf{x}) \quad (12)$$

Our definition for membership functions is different from that proposed by [2, 3]. The difference comes from our interpretation of the margin as an area of uncertainty regarding the output the SVM offers. You can only have certainty for the answer, when the decision function value is out of the margin, namely when $|f_{ij}(\mathbf{x})| \geq 1$. The band is an area where the uncertainty on the output computed by the SVM grows as $f_{ij}(\mathbf{x})$ moves towards the opposite subspace (Fig. 1). We think that a negative value for $f_{ij}(\mathbf{x})$ does not mean a zero degree of \mathbf{x} belonging to class i . In this case, the membership degree should be small — actually, less than 0.5. But it will only extinguish when going over the opposite band frontier. A

value of 0 for $f_{ij}(\mathbf{x})$ is a completely uncertain answer, hence $\mu_{ij}(\mathbf{x}) = 0.5$.

To obtain the (overall) membership degree of \mathbf{x} for each class, we need to aggregate $(k - 1)$ membership degrees somehow. Fuzzy Logic literature is rich in aggregation operators, as can be easily checked through a quick browse of publications. We have considered several choices, but we will only show results for four of them: the average operator, the min operator, the harmonic average and the OWA operator [16].

The OWA operators deserve special attention. They have an important feature: they associate a weight with a position with respect to argument relative order. The OWA operator requires a non-increasing sort of the arguments, which makes it a nonlinear operator. Then every position is pondered by a weight, which can be learned. Yager's original proposal was to learn those weights for specific applications through a gradient descent method [17]. This adaptability of the operator endows the system further flexibility allow it to better reproduce the expected fuzzy output.

The final (fuzzy) answer of the system is expressed in terms of k pairs $(i, m_i(\mathbf{x}))$, with $m_i(x)$ the membership degree of \mathbf{x} to the class i . $m_i(x)$ is computed by aggregating the $k - 1$ different membership functions for the class i .

When a crisp decision is necessary, the class assigned to \mathbf{x} is

$$\text{class} = \arg \max_{i=1, \dots, k} m_i(x) \quad (13)$$

4 Empirical Analysis

To test the validity of our proposal as well as its performance, we have conducted a number of experiments. Basically, they imply building the model described above for a number of well known problems. Then the model performance is compared against non fuzzy versions of SVMs.

4.1 Methodology

The selected benchmark datasets are: **dna**, **satimage**, **glass** and **thyroid**. The two first datasets are included in Statlog collection [18] and

the other datasets belongs to UCI [19]. For **dna**, **satimage**, and **thyroid** datasets, there are already standard splits in train and test parts. To make fair use of the data, we applied a 10-fold cross-validation process —on the training set— to conduct model selection. Then we used all the training data to build the model and evaluated its performance on the test set. As for the **glass** dataset, we have applied directly a 10-fold cross-validation to evaluate the model. Hence, the performance result showed for **glass** is the average of all folds.

The model selection has been conducted as follows: To minimize the parameter set, we trained every dataset only with the RBF kernel $K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$. For the first three problems, we estimated the generalized performance using $\sigma^2 = \{\frac{1}{2^\gamma}\}$, where $\gamma = \{2, \dots, 2^{-7}, 2^{-8}\}$ and $C = \{2, 2^2, \dots, 2^7\}$. After these experiments, we tried to tune kernel and cost parameters through some trial-and-error. In summary, we made $10 \times 7 \times 3 = 210$ experiments. As for the **thyroid** dataset, SVMs with the following parameter sets were trained: $\sigma = \{1.0\}$ and $C = \{1, 10, 10^2, \dots, 10^4\}$. Those values were taken from [3]. So, we made 5 experiments. To sum up, we have run $215 \times 10 = 2150$ computational experiments. Finally, we stop the optimization algorithm for each problem, if the KKT violation is less than 10^{-3} .

Detailed information regarding the final winning parameter sets is displayed on Table 1.

4.2 Results

Results for our experiments are shown on Table 2. These results are complemented with performance of conventional (crisp) SVM for both approaches Max-Win and DAG, which are published somewhere else. Figures for **dna**, **satimage** and **glass** are taken from [20]. We also computed the datum for the **thyroid** dataset.

These results prove that the fuzzy classifier built on the SVM does not loose any precision compared against the crisp models. There are slight variations in results in both senses, but so small, that they are not statistically significant. In ad-

dition, these results show a little improvement over those obtained with the other proposal for building fuzzy classifiers with SVMs, the fuzzy LS-SVMs [2, 3].

5 Conclusions

We have proposed a way to build a classifier with fuzzy outputs based on a pairwise multiclass Support Vector Machine. This proves the applicability of this sound founded model to a new class of problems. This points in a way in which the strong and rigorous foundations of Statistical Learning Theory can be used to back up results obtained with fuzzy systems.

We have conducted a number of experiments to test the performance of the new model applying it on several real world datasets. The empirical results show that there is no loss of performance, and the expressiveness is highly improved.

Acknowledgments

This research has been partially supported by Ministerio de Ciencia y Tecnología under projects TIC2003-04650 and TIN2004-07236.

References

- [1] I.M. Guyon B.E. Boser and V. Vapnik. A training algorithm for optimal margin classifiers. In Fifth Annual Workshop on Computational Learning Theory, ACM, 1992.
- [2] Daisuke Tsujinishi and Shigeo Abe. Fuzzy least squares support vector machines for multiclass problems. *Neural Networks*, 16(5-6):785–792, 2003.
- [3] Yoshiaki Koshiha Daisuke Tsujinishi and Shigeo Abe. Why pairwise is better than one-against-all or all-at-once. volume 1, pages 693–698. IEEE International Conference on Neural Networks - Conference Proceedings 1, IEEE Press, 2004.
- [4] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16:264–280, 1971.
- [5] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons Inc. New York, 1998.
- [6] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, Massachusetts. London, England., 2002.
- [7] Nello Cristianini and John Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, The Edinburgh Building Cambridge CB2 2RU, UK, third edition, 2003.
- [8] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. Kluwer Academic Publishers, Boston., 2001. Bell Laboratories, Lucent Technologies.
- [9] J. Weston and C. Watkins. Multi-class support vector machines. Proc. ESANN99, M. Verleysen Ed., Brussels, Belgium, 1999.
- [10] V. Vapnik. Estimation of dependences based on empirical data [in russian], 1979. (English Translation: Springer Verlag, New York, 1982).
- [11] R. Freund E. Osuna and F. Girosi. An improved training algorithm for support vector classification. pages 276–285. Proc. IEEE Neural Networks for Signal Processing VII Workshop, IEEE Press, 1997.
- [12] John C. Platt. Fast training of support vector machines using sequential minimal optimization. The MIT Press, Massachusetts, 1999. Advances in Kernel Methods, Chapter 12.
- [13] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *Comput. Learning Theory*, pages 35–46, 2000.
- [14] Ulrich H.-G. Krebel. Pairwise classification and support vector machines. The MIT Press, Massachusetts, 1999. Advance in Kernel Methods, Chapter 15.
- [15] Nello Cristianini John C. Platt and John Shawe-Taylor. Large margin dags for multiclass classification. *Advances in Neural Information Processing System*, 12:547–553, 2000.

	<i>Max-Win</i>	<i>DAG</i>	<i>Avg</i>	<i>Min</i>	<i>OWA</i>	<i>Harmonic</i>
<i>Promoters</i>	(2 ³ , 5.65)	(2 ³ , 5.65)	(2.5, 5.66)	(2 ⁴ , 5.66)	(1.8, 5.66)	(1.8, 5.66)
<i>Satimage</i>	(2 ⁴ , 0.71)	(2 ⁴ , 0.71)	(2 ² , 0.50)	(3, 0.71)	(3, 0.71)	(3.0, 0.71)
<i>Glass</i>	(2 ¹¹ , 1.41)	(2 ¹² , 2 ¹)	(40, 2 ⁰)	(75, 2 ⁰)	(40, 2 ⁰)	(75, 2 ⁰)
<i>Thyroid</i>	(10 ⁴ , 2 ⁰)	(10 ⁴ , 2 ⁰)	(10 ⁴ , 2 ⁰)	(10 ⁴ , 2 ⁰)	(10 ⁴ , 2 ⁰)	(10 ⁴ , 2 ⁰)

Table 1: (C, σ) parameter set.

	<i>Max-Win</i>	<i>DAG</i>	<i>Avg</i>	<i>Min</i>	<i>OWA</i>	<i>Harmonic</i>
<i>Promoters</i>	95.45	95.45	95.70	95.45	95.86	95.86
<i>Satimage</i>	91.30	91.25	90.80	90.80	90.70	90.80
<i>Glass</i>	71.50	73.85	72.85	73.80	72.85	72.85
<i>Thyroid</i>	97.08	97.08	97.14	97.19	97.02	97.17

Table 2: Experimental results. Best test rates bold-faced.

- [16] R.R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(1):183–190, 1988.
- [17] Dimitar Filev and Ronald R. Yager. Learning owa operator weights from data. volume 1, pages 468–473. IEEE International Conference on Fuzzy Systems, IEEE Press, 1994.
- [18] D. J. Spiegelhalter D. Michie and C. C. Taylor (1994). Machine learning, neural and statistical classification. [Online] <http://www.niaad.liacc.up.pt/statlog/>.
- [19] C. L. Blake and C. J. Merz. (1998). UCI Repository of machine learning databases. univ. california, dept. inform. comput. sci., irvine, ca. [Online] <http://www.ics.uci.edu/mlearn/MLSummary.html>.
- [20] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, March 2002.