Fuzzy-Input Fuzzy-Output One-Against-All Support Vector Machines

Christian Thiel, Stefan Scherer and Friedhelm Schwenker

Institute of Neural Information Processing, University of Ulm, 89069 Ulm, Germany
christian.thiel|stefan.scherer|friedhelm.schwenker@uni-ulm.de

Abstract. We present a novel approach for Fuzzy-Input Fuzzy-Output classification. One-Against-All Support Vector Machines are adapted to deal with the fuzzy memberships encoded in fuzzy labels, and to also give fuzzy classification answers. The mathematical background for the modifications is given. In a benchmark application, the recognition of emotions in human speech, the accuracy of our F^2-SVM approach is clearly superior to that of fuzzy MLP and fuzzy K-NN architectures.

1 Introduction

Support Vector Machines (SVMs) have become a popular method in pattern classification and were originally developed for the discrimination of two-class problems [1]. They work by projecting the data into a higher dimensional feature space using kernel functions, then finding the hyperplane that separates the two classes while providing the widest margin in which no sample points lie (see [2] for an introduction). Later, considerable research has been carried out to find ways to make use of the principle of SVMs in multi class problems (see recently [3]), and architectures like One-Against-One, DAG-SVMs or One-Against-All are widely used nowadays (see [4] for a comparison).

The previously mentioned approaches work with data that features hard, or "crisp", labels, that is, each training sample belongs to exactly one class. Now one could easily imagine situations where the data is not labeled in that way, for example when the labelling expert is given the choice to spread his opinion over multiple classes, or to give an indication how certain he is in his decision. The latter case has now been addressed by Lin and Wang [5] and Huang and Liu [6], although both articles developed the method with a different goal in mind: their fuzzy memberships respectively membership values are not given, but constructed from the hard labeled data to solve special problems, here the weighting of samples in time series or the decrease of the impact of (detected) outliers in the data. The outputs of those SVMs however remain hard, they "produce an uncalibrated value that is not a probability". Having observed this, Platt [7] developed a method that transforms the observed distances using a parametric form of a sigmoid function. The parameters are fit using maximum likelihood estimation on the training set.

So far, those SVMs can not be put to work on the task of fuzzy-fuzzy classification where the training data carries only soft labels, meaning each sample

point can be assigned to multiple classes in varying degrees, and such a soft output is also expected from the algorithm. Motivated by a real-world application, the fuzzy classification of emotion in recordings of spoken sentences, we developed a method that will accomplish this fuzzy-fuzzy classification with the use of Support Vector Machines. Very recently, a similar method was presented by Borasca et al. [8], although our approach to deal with the multi class problem is much less computationally expensive. But we would like to take up their naming and call the fuzzy-input fuzzy-output Support Vector Machine **$\mathbf{F^2}$-SVM** in the following.

The F^2-SVMs will be derived and explained in the following section, then in section 3 the emotion-recognition application is presented and experiments reported that compare the performance of our method to more established ones. Some interesting findings will be highlighted. Then, in section 4, a promising venue of further research using One-Class SVMs will be presented. In the closing statement, we explain why it is well worth to use the F^2-SVM.

2 Fuzzy-Input Fuzzy-Output SVMs

2.1 Basic SVMs

As a foundation for the following introduction of our F^2-SVMs, we will briefly review the basic theory behind Support Vector Machines (see also in [2] or [9]).

As mentioned above, SVMs were developed to solve two-class problems. That is, there is a training set S given as

$$S = \{(x_\mu, l_\mu\ )|\ \mu = 1, \ldots, M,\ x_\mu \in \mathbb{R}^N,\ l_\mu \in \{-1, +1\}\ \}, \tag{1}$$

which can be divided up into two sets by a separating hyperplane. Such a hyperplane is determined by a weight vector $w \in \mathbb{R}^N$ and a bias or threshold $w_0 \in \mathbb{R}$ satisfying the separating constraints

$$l_\mu\ \ (x_\mu^T w + w_0) \geq 1, \quad \mu = 1, \ldots, M. \tag{2}$$

The hyperplane has a margin on both sides within which no training points lie. The idea now is to maximise this margin of width $2/||w||$, which leads to the following minimisation problem that is subject to the constraints just mentioned (2):

$$\Theta(w) = w^T w/2 \rightarrow \min \tag{3}$$

But, if the constraints do not hold for some data points, the problem is not linearly separable, and we have to soften them by introducing slack variables ξ_μ (this is called the "soft margin" approach). For some data points, it is now permissible to lie within the margin ($0 \leq \xi_\mu < 1$) or even on the wrong side of the hyperplane ($\xi_\mu > 1$). The optimisation problem and the constraints become:

$$\Theta(w, \xi) = w^T w/2 + C \sum_{\mu=1}^{M} \xi_\mu \rightarrow \min \tag{4}$$

$$l_\mu \ (x_\mu^T w + w_0) \geq 1 - \xi_\mu, \quad \xi_\mu \geq 0, \quad \mu = 1, \ldots, M \tag{5}$$

Note the free parameter $C > 0$ which regulates the amount of margin-violations the SVM has to tolerate in finding the optimal hyperplane.

A very important feature of SVMs is the use of so-called kernel functions K. A Mercer kernel function (see [1]) implicitly transforms the data points of the input space to a high dimensional Hilbert space, where now it might be possible to find a separating hyperplane, if we do not manage to do so in the original input space. Calculating the dot-product in Hilbert space using the kernel function, is the so-called kernel trick:

$$\langle \phi(x_1), \phi(x_2) \rangle = K(x_1, x_2), \quad x_1, x_2 \in \mathbb{R}^N \tag{6}$$

The transformation function ϕ from input space to Hilbert space does not need to be evaluated, since the dot-product is given implicitly by the chosen kernel function K.

2.2 Deriving Fuzzy-Input SVMs

Now we want to make use of data where samples are not hard labeled, but associated with multiple classes each. As the basic SVM architecture is specialised on discriminating between two classes, we rest in the binary case for now, the extension to any desired number of classes will be explained in section 2.3. In our approach there are two membership values, m_μ^- and m_μ^+, associated with each training sample $x_\mu \in S$. These values indicate to what extent the sample point belongs to each of the two classes $\{-1, +1\}$. These memberships are incorporated into our minimisation problem (4) by weighting the importance of the error-indicating slack variables ξ_μ accordingly:

$$\Theta(w, \xi^+, \xi^-) = w^T w/2 + C \sum_{\mu=1}^{M} (\xi_\mu^+ m_\mu^+ + \xi_\mu^- m_\mu^-) \rightarrow \min \tag{7}$$

For the fuzzy SVM, a separating hyperplane has to be calculated under the following constraints:

$$w^T x_\mu + w_0 \geq 1 - \xi_\mu^+, \qquad \mu = 1, \ldots, M \tag{8}$$

$$w^T x_\mu + w_0 \leq -(1 - \xi_\mu^-), \qquad \mu = 1, \ldots, M \tag{9}$$

$$\xi_\mu^+ \geq 0 \quad \text{and} \quad \xi_\mu^- \geq 0, \qquad \mu = 1, \ldots, M \tag{10}$$

Because this primal problem is very hard to solve with quadratic programming, we introduce the Lagrange multipliers α^+, α^- and β^+, β^- for our constraints,

so that now the problem becomes finding the saddle point of the Lagrangian L:

$$
\begin{aligned}
L(w, w_0, \xi^+, \xi^-, \alpha^+, \alpha^-, \beta^+, \beta^-) &= ww^T/2 + C\sum_{\mu=1}^{M}(\xi_\mu^+ m_\mu^+ + \xi_\mu^- m_\mu^-) \\
&- \sum_{\mu=1}^{M}\alpha_\mu^+((w^T x_\mu + w_0) - (1-\xi_\mu^+)) + \sum_{\mu=1}^{M}\alpha_\mu^-((w^T x_\mu + w_0) + (1-\xi_\mu^-)) \quad (11) \\
&- \sum_{\mu=1}^{M}\beta_\mu^+\xi_\mu^+ - \sum_{\mu=1}^{M}\beta_\mu^-\xi_\mu^-
\end{aligned}
$$

Differentiating L with respect to the variables w, w_0, ξ^+, ξ^- of our primal optimisation problem, and setting the resulting terms equal to zero, we obtain the following necessary conditions:

$$\frac{\partial L}{\partial w} = w - \sum_{\mu=1}^{M}\alpha_\mu^+ x_\mu + \sum_{\mu=1}^{M}\alpha_\mu^- x_\mu = 0 \quad \Rightarrow \; w = \sum_{\mu=1}^{M}(\alpha_\mu^+ - \alpha_\mu^-)x_\mu \qquad (12)$$

$$\frac{\partial L}{\partial w_0} = -\sum_{\mu=1}^{M}\alpha_\mu^+ + \sum_{\mu=1}^{M}\alpha_\mu^- = 0 \quad \Rightarrow \; \sum_{\mu=1}^{M}(\alpha_\mu^+ - \alpha_\mu^-) = 0 \qquad (13)$$

$$\frac{\partial L}{\partial \xi_\mu^+} = Cm_\mu^+ - \alpha_\mu^+ - \beta_\mu^+ = 0, \quad \frac{\partial L}{\partial \xi_\mu^-} = Cm_\mu^- - \alpha_\mu^- - \beta_\mu^- = 0 \qquad (14)$$

Inserting (12) into (11), multiplying out and reordering yields

$$
\begin{aligned}
L = & \underbrace{-\sum_{\mu=1}^{M}\alpha_\mu^+ w_0 + \sum_{\mu=1}^{M}\alpha_\mu^- w_0}_{=0 \text{ because of (13)}} \\
&+ \sum_{\mu=1}^{M}\alpha_\mu^+(1-\xi_\mu^+) + \sum_{\mu=1}^{M}\alpha_\mu^-(1-\xi_\mu^-) + C\sum_{\mu=1}^{M}(\xi_\mu^+ m_\mu^+ + \xi_\mu^- m_\mu^-) - \sum_{\mu=1}^{M}\beta_\mu^+\xi_\mu^+ - \sum_{\mu=1}^{M}\beta_\mu^-\xi_\mu^- \\
&- \sum_{\mu=1}^{M}\sum_{\nu=1}^{M}\alpha_\mu^+(\alpha_\nu^+ - \alpha_\nu^-)x_\nu^T x_\mu + \sum_{\mu=1}^{M}\sum_{\nu=1}^{M}\alpha_\mu^-(\alpha_\nu^+ - \alpha_\nu^-)x_\nu^T x_\mu \\
&+ 1/2\sum_{\mu=1}^{M}\sum_{\nu=1}^{M}(\alpha_\mu^+ - \alpha_\mu^-)(\alpha_\nu^+ - \alpha_\nu^-)x_\mu^T x_\nu \; .
\end{aligned}
\qquad (15)
$$

Simplifying the quadratic parts at the end and reordering yields

$$L = -1/2 \sum_{\mu=1}^{M} \sum_{\nu=1}^{M} (\alpha_\mu^+ - \alpha_\mu^-)(\alpha_\nu^+ - \alpha_\nu^-) x_\mu^T x_\nu$$
$$+ \sum_{\mu=1}^{M} (\alpha_\mu^+ - \underbrace{\alpha_\mu^+ \xi_\mu^+ + C\xi_\mu^+ m_\mu^+ - \beta_\mu^+ \xi_\mu^+}_{=0 \text{ because of (14)}}) + \sum_{\mu=1}^{M} (\alpha_\mu^- - \underbrace{\alpha_\mu^- \xi_\mu^- + C\xi_\mu^- m_\mu^- - \beta_\mu^- \xi_\mu^-}_{=0 \text{ because of (14)}}) . \tag{16}$$

According to the Karush-Kuhn-Tucker theory, with $\alpha_\mu^+, \alpha_\mu^-, \beta_\mu^+, \beta_\mu^- \geq 0$, the dual problem is now to maximise

$$L(\alpha) = \sum_{\mu=1}^{M} \alpha_\mu^+ + \sum_{\mu=1}^{M} \alpha_\mu^- - 1/2 \sum_{\mu=1}^{M} \sum_{\nu=1}^{M} (\alpha_\mu^+ - \alpha_\mu^-)(\alpha_\nu^+ - \alpha_\nu^-) x_\mu^T x_\nu \tag{17}$$

with the product $x_\mu^T x_\nu$ at the end calculated using a kernel function (6), and subject to

$$\sum_{\mu=1}^{M} (\alpha_\mu^+ - \alpha_\mu^-) = 0 \quad \text{from (13) and} \tag{18}$$

$$0 \leq \alpha_\mu^+ \leq C m_\mu^+, \quad 0 \leq \alpha_\mu^- \leq C m_\mu^- \quad \text{from (14).} \tag{19}$$

The difference to ordinary SVMs is that we have doubled the number of sample points m_μ, by having each as positive and negative sample, and that each Lagrange multiplier α_μ is now not bounded simply by the fix, a priori set C, but by a function (19) that takes into account the membership for each point. For less important samples, α_μ has now a smaller range to be selected from.

The Karush-Kuhn-Tucker conditions for the problem now are (with $\mu = 1, \ldots, M$):

$$\alpha_\mu^+ ((w^T x_\mu + w_0) - (1 - \xi_\mu^+)) = 0, \quad \alpha_\mu^- ((w^T x_\mu + w_0) + (1 - \xi_\mu^-)) = 0 \tag{20}$$

$$\beta_\mu^+ \xi_\mu^+ \underset{(14)}{=} (C m_\mu^+ - \alpha_\mu^+) \xi_\mu^+ = 0, \quad \beta_\mu^- \xi_\mu^- \underset{(14)}{=} (C m_\mu^- - \alpha_\mu^-) \xi_\mu^- = 0 \tag{21}$$

Those samples x_μ associated with a combined Lagrange multiplier $\alpha_\mu = (\alpha_\mu^+ - \alpha_\mu^-) \neq 0$ are the important Support Vectors SV, which determine the separating hyperplane (compare with condition 12):

$$w = \sum_{\mu=1}^{M} (\alpha_\mu^+ - \alpha_\mu^-) x_\mu \tag{22}$$

Support Vectors x_μ with $\alpha_\mu^+ = m_\mu^+ C$ or $\alpha_\mu^- = m_\mu^- C$ will be situated, according to (21), within the margin or even beyond the separating hyperplane. With increasing association of samples to multiple classes, this should be expected to happen quite frequently.

Now, using an on-the-margin Support Vector in (20) to obtain w_0, the final decision function f to classify a new sample z is

$$f(z) = sign(w^T z + w_0) \; . \tag{23}$$

Again, all samples are projected into the higher-dimensional Hilbert space H using a kernel function K. In this case the final decision function is (see (22) and (6)):

$$f(z) = sign(\sum_{i \in SV} (\alpha_i^+ - \alpha_i^-) \; K(z, x_i) + w_0^H) \tag{24}$$

So far, the fuzzy-input Support Vector Machines only deal with two classes at a time. To extent this to the multi class case, with k classes, we will be using the One-Against-All architecture. It works by building k different SVM_i, each of which is capable of separating one class i from all others. As training data, we still have our training set $S = \{(x_\mu, l_\mu) | \mu = 1, \ldots, M\}$, still with $x_\mu \in \mathbb{R}^N$, but now

$$l_\mu = (l_{1,\mu}, l_{2,\mu}, \ldots, l_{k,\mu}), \;\; l_{i,\mu} \in [0,1], \;\; \sum_{i=1}^{k} l_{i,_\mu} = 1. \tag{25}$$

That is, each sample x_μ now belongs to a different degree to each of the k classes. The training data S_i^{train} for a SVM_i is now constructed by taking all samples points twice, as explained at the beginning of this section, only using a part of the label, l_i, to form the membership values $m_{i,\mu}$:

$$S_i^{train} = \{(x_\mu, m_{i,\mu}^+) | m_{i,\mu}^+ = l_{i,\mu}\} \cup \{(x_\mu, m_{i,\mu}^-) | m_{i,\mu}^- = 1 - l_{i,\mu}\}, \quad \mu = 1, \ldots, M \tag{26}$$

Each of the SVM_i is now trained. How their outputs to a sample z are again transformed into the appropriate estimation for l_μ is covered in the next section.

At this point, it is appropriate to mention the differences between our One-Against-All approach and the One-Against-One architecture used in [8]. In the latter case, it is necessary to build $k \; (k-1)/2$ Support Vector Machines, that each distinguish between two classes, while with our approach, we have k machines. The number of samples used to train the individual machines is the same for both strategies, 2 M, so our fuzzy One-Against-All approach will be considerably faster for problems with more than three classes. On the other hand, it will have to be determined experimentally if the bigger One-Against-One architecture could not yield a higher accuracy with very complex data sets.

2.3 Obtaining the Fuzzy-Output

Now suppose we have trained the fuzzy SVM architecture as described above. If we feed a new sample z to the $SVM_i, i = 1, \ldots, k$, the result will be values $d_i \in \mathbb{R}$, one from each of the One-Against-All SVMs. These d_i represent the distances in kernel space of the sample z to the separating hyperplanes determined for the machines. To transform them into fuzzy output labels o_i, we make use of a sigmoid function, as recommended in [7]:

$$o_i(z) = 1/(1 + \exp(-A_i^T z + B_i)), \quad i = 1, \ldots, k \tag{27}$$

The parameters $A_i \in \mathbb{R}^N$ and $B_i \in \mathbb{R}$ are estimated for each SVM_i to minimise the mean squared error on the training data S_i^{train} between the original label and the sigmoid output:

$$error_i = \frac{1}{M} \sum_{x_\mu \in S_i^{train}} (o_i(x_\mu) - l_{i,\mu})^2 \tag{28}$$

Estimation is accomplished via a batch gradient descent technique that stops if there are only slight adjustments of the parameters between iterations. The resulting update rules with learning rate η are:

$$\begin{aligned} \Delta A_i &= (o_i(x_\mu) - l_{i,\mu})\ \eta\ o_i^2(x_\mu)(-x_\mu) \\ \Delta B_i &= (o_i(x_\mu) - l_{i,\mu})\ \eta\ o_i^2(x_\mu) \end{aligned} \tag{29}$$

Note that the fuzzy output labels o_i are normalised to sum up to 1.

3 Experiments and Conclusions

The aim of our experiments was to evaluate the performance of our F^2-SVMs and compare it against other established fuzzy classification methods, expecially fuzzy K-Nearest-Neighbour (KNN) and fuzzy Multi-Layer Perceptron (MLP). But to be thorough, we also included comparisons with standard hard SVMs.

As application, we chose the detection and classification of emotions in human speech, where it is very natural to have multiple emotions to varying degrees at the same time, motivating our use of fuzzy labels. The scenario is a setting where two humans carry on a conversation in front of a computer, with its special task being to give restaurant recommendations. The utilised speech corpus has been recorded within the project 'Perception and Interaction in Multi-User Environments' [10] at the competence centre Perception and Interactive Technologies (PIT). This corpus contains over 400 German utterances which are basically short sentences recorded from 4 different speakers (1 female and 3 male). The enacted emotions are hot-anger, happiness, fear, sadness, disgust, boredom and neutral. Sadness and fear were omitted from most experiments, since they are not necessary in our application setting. Sentences for producing the database were collected from everyday communications with semantically neutral context. Complementary, a human emotion recognition performance test has been conducted on the data, not only as a benchmark for the automatic emotion recognisers, but also for the fuzzy-labelling of the sentences. The answers of 10 test persons were recorded for each utterance and after aggregation and normalisation then formed the soft labels for our data.

Many researchers have already investigated the acoustic cues of emotions in speech which have predictable effects on speech, especially on pitch, timing and voice quality (see for example [11] or [12]). To obtain the features for our classification example, each sentence of the corpus was analysed based on a time window of a length of 30ms with an overlap of 20ms. The results of the analysis yielded different parameters regarding the acoustics of the uttered sentences

such as pitch (fundamental frequency, computed using the SIFT algorithm [13] which performed satisfactory), the first derivative of the pitch, energy and the first three formant contours. From each parameter, statistical features such as mean, percentiles, maximum and minimum, were extracted. Other characteristics, for example voiced and unvoiced segments and the temporal aspects of the utterances, were statistically analysed as well, leading to a total of 37 scalar statistical features for each utterance.

We shall be giving some details on our experimental setup: For all of our experiments, we used 10 times 10 fold cross validation. The SVMs were trained using the SMO algorithm by Platt [14]. The fuzzy KNN considers the fuzzy labels of the 5 nearest samples in the training data, sums their labels and normalises them. The fuzzy MLP has 30 neurons in its hidden layer, minimising the mean square error between network output and the fuzzy labels using backpropagation over 20 epochs (more iterations did not decrease the error significantly).

	SVM	Fuzzy MLP	Fuzzy KNN	F^2-SVM
Accuracy (in %)	54.1	33.1	29.5	**55.6**
Euclidean distance	0.566	2.132	0.724	**0.541**
S_1 Min/Max	0.361	0.141	0.248	**0.395**

Table 1. Performance of the different classifiers. The accuracy is measured against defuzzified soft labels. The Euclidean distance is measured against soft labels, as well as the S_1 similarity measure. (S_1 was defined in [15] as $S_1(A,B) = \| A \cap B \| \, / \, \| A \cup B \|$.) Our F^2-SVM algorithm does clearly outperform the non-SVM algorithms.

In our experiments, we made several interesting observations. For one, it was straightforward to chose a polynomial kernel function K with a degree of three. The linear and RBF kernels did never reach a comparable performance, and the RBF kernel was also very sensitive to changes of its width parameter.

As could be expected, the choice of C was crucial for the performance of the classifier. If this were not the case, we could not hope to get better results by modifying the weight of individual samples in the error term via memberships (see (7)). Using cross validation experiments, we determined the optimal values for C in the standard- respectively F^2-SVMs to be 10^{-3} and 10^{-2}, out of a range of $[10^{-4}; 10^{3}]$. New experiments [16] suggest that, at least for two-class problems, the choice of C is not important for very high-dimensional problems.

One of the most important questions was wether the F^2-SVMs would yield a better classification rate on the defuzzified[1] decisions than SVMs trained with defuzzified labels. This turned out to be the case, pairwise experiments showed that the accuracy of the F^2-SVMs was 1.55 percent points higher on average, out of the 100 runs it lost only in 30, a tie was reached in 16 runs. This shows that

[1] Defuzzification here means that the fuzzy labels or decisions were converted using the maximum rule, hence they would each indicate one winner-class only.

even when a hard final output is necessary, it is beneficial to use the F^2-SVMs trained with fuzzy labels.

For our data, we did not only have the soft labels produced by multiple humans giving their opinion on the emotion expressed in each sentence, but also hard labels with the emotion the speakers were told to express. The defuzzified soft labels coincide only in 80% of the cases with the original hard labels. Now, training with the soft labels, and comparing the defuzzified final outputs to the two kinds of test labels, defuzzified fuzzy and original hard, it turned out that the performance on both was about the same. The accuracy against the original hard labels was even slightly higher. So, the classifier has never seen the original hard labels, which are somewhat different to the soft labels, but delivers an equal accuracy on them. This is a strong indication that the fuzzy labels really help the SVMs in capturing the emotion distribution.

4 Future approaches

Another kind of Support Vector Machine lends itself to be adapted for fuzzy-in fuzzy-out classification, the so-called One-Class SVMs. They work by not finding a separating hyperplane, but fitting all data samples from one class within one circle (in kernel space, of course) with radius R. Again we can, analogue to equation 7, introduce a weighting of the slack variables ξ_μ with the membership m_μ of the samples:

$$\Theta(R, \xi) = R^2 + C \sum_{\mu=1}^{M} \xi_\mu m_\mu \rightarrow \min \tag{30}$$

Construction the Lagrangian and differentiating it with respect to ξ_μ yields the following condition, analogue to equation 14:

$$C m_\mu - \alpha_\mu - \beta_\mu = 0 \tag{31}$$

This is a nice result and means that the One-Class SVMs can really be adapted to use fuzzy class labels, still allowing optimisation via the SMO algorithm. We plan to pursue this venue in future research.

5 Summary

We proposed a new method to deal with fuzzy labels in classification, making good use of the power that kernels provide. Our F^2-SVMs perform better than standard hard-trained SVMs in pairwise experiments and have a much higher accuracy than fuzzy MLP or fuzzy K-NN classifiers. Unlike earlier approaches, we do not set a certainty value for each sample, but use memberships that allow each sample to be associated with multiple classes to a different degree. The SVM training formula we derived is analogue to standard SVMs, hence they and the One-Against-All architecture we used could be adapted to our approach without major problems. The F^2-SVMs are full fuzzy-in fuzzy-out classifiers.

6 Acknowledgements

This work was partially supported by the DFG (German Research Society) contract SCHW 623/3-2.

References

1. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer (1995)
2. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press (2002)
3. Angulo, C., Ruiz, F.J., González, L., Ortega, J.A.: Multi-Classification by using Tri-Class SVM. Neural Processing Letters **23** (2006) 89–101
4. Kahsay, L., Schwenker, F., Palm, G.: Comparison of multiclass SVM decomposition schemes for visual object recognition. In: DAGM 2005. Volume 3663 of LNCS., Springer (2005) 334–341
5. Lin, C.F., Wang, S.D.: Fuzzy Support Vector Machines. IEEE Transactions on Neural Networks **13** (2002) 464–471
6. Huang, H.P., Liu, Y.H.: Fuzzy Support Vector Machines for Pattern Recognition and Data Mining. International Journal of Fuzzy Systems **4** (2002) 826–835
7. Platt, J.C.: Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In: Advances in Large Margin Classifiers, NIPS 1998, MIT Press (1999) 61–74
8. Borasca, B., Bruzzone, L., Carlin, L., Zusi, M.: A Fuzzy-input Fuzzy-output SVM Technique for Classification of Hyperspectral Remote Sensing Images. In: Proceedings of the NORSIG 2006, Reykjavík. (2006)
9. Webb, A.R.: Statistical Pattern Recognition. second edn. John Wiley & Sons (2002)
10. Strauss, P.M., Hoffmann, H., Minker, W., Neumann, H., Palm, G., Scherer, S., Schwenker, F., Traue, H., Walter, W., Weidenbacher, U.: Wizard-of-oz data collection for perception and interaction in multi-user environments. In: International Conference on Language Resources and Evaluation (LREC). (2006)
11. Banse, R., Scherer, K.R.: Acoustic profiles in vocal emotion expression. Journal of Personality and Social Psychology **70** (1996) 614–636
12. Dellaert, F., Polzin, T., Waibel, A.: Recognizing emotion in speech. In: Proceedings of the ICSLP. (1996) 1970–1973
13. Rabiner, L.R., Schafer, R.W.: Digital Processing of Speech Signals. Prentice-Hall Signal Processing Series (1978)
14. Platt, J.C.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technical Report 98-14, Microsoft Research, Redmond, Washington (1998)
15. Dubois, D., Prade, H.: Fuzzy Sets and Systems: Theory and Applications. Academic Press (1980)
16. Tegnér, J.: Evaluating Feature Selection for SVMs in High Dimensions. In: Proceedings of the 17th European Conference on Machine Learning. Volume 4212 of LNAI., Springer (2006) 719–726