Computer Vision and Image Understanding xxx (2009) xxx–xxx

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

An optimization on pictogram identification for the road-sign recognition task using SVMs

S. Maldonado Bascón [a,*], J. Acevedo Rodríguez [a], S. Lafuente Arroyo [a], A. Fernndez Caballero [b,c], F. López-Ferreras [a]

[a] Departamento de Teoría de la Seal y Comunicaciones, Escuela Politécnica Superior, Universidad de Alcalá, 28871 Alcalá de Henares, Madrid, Spain
[b] Instituto de Investigación en Informática de Albacete (I3A), Universidad de Castilla-La Mancha, 02071 Albacete, Spain
[c] Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, 02071 Albacete, Spain

ARTICLE INFO

Article history:
Received 18 April 2008
Accepted 1 December 2009
Available online xxxx

Keywords:
Road sign
Automatic traffic sign detection and recognition system (TSDRS)
Classification
Support vector machines (SVMs)

ABSTRACT

Pattern recognition methods are used in the final stage of a traffic sign detection and recognition system, where the main objective is to categorize a detected sign. Support vector machines have been reported as a good method to achieve this main target due to their ability to provide good accuracy as well as being sparse methods. Nevertheless, for complete data sets of traffic signs the number of operations needed in the test phase is still large, whereas the accuracy needs to be improved. The objectives of this work are to propose pre-processing methods and improvements in support vector machines to increase the accuracy achieved while the number of support vectors, and thus the number of operations needed in the test phase, is reduced. Results show that with the proposed methods the accuracy is increased 3–5% with a reduction in the number of support vectors of 50–70%.

1. Introduction

The visibility of traffic signs is crucial for the drivers and pedestrians safety [1]. For this reason, automatic systems developed to give information about traffic signs are one of the most important issues for research of intelligent transportation systems. Although there are some systems focused in the retroreflectivity [2], most of the research in this field has been done in automatic traffic sign detection and recognition systems (TSRDS) with a wile range of applications to be applied: autonomous driving, driving assistance systems and automatic road sign inventory.

Most of TSDRS are divided into three stages: segmentation, detection and pictogram classification. The role of the first stage is to isolate candidate blobs to be classified, whereas the detection block selects those blobs that have an appropriate traffic sign shape and the classification stage identifies the information of the traffic sign or determine that the candidate blob is noise.

There are many research fields to improve TSDRS such as improvements in the segmentation and detection blocks. Nevertheless, one of the most important issues is to improve the classification stage in such a way that recognition time can be minimized while keeping a high accuracy level. Generally, the input to this stage is variable sized and candidate regions must be appropriately represented for a given classification technique. There are two options to make the classification:

1. To extract some features from the image and to use these features as inputs to the classifier.
2. To present the part of the image itself through the sub-sampled pixel intensities.

In the first case, each candidate blob is characterized by a vector. Examples of methods to extract features used in road sign identification are histograms [3], FFT computed after a complex log-mapping of exterior borders [4,5] and wavelets [6,7].

In the second approach, among those who work directly with the image, different sizes are considered. The compromise between a good resolution to discern the classes and computational cost determines the size to which the images are scaled. Some examples of normalized images are 18×18 pixels in. [8], 30×30 in. [9], 16×16 in. [10] and 31×31 in. [11]. Additionally, in [11,12] a mask is applied before the recognition task in order to remove the background regarding the information of the sign board shape.

Having adopted the appropriate features, several recognition methods have been proposed such as a Back-propagation Neuronal Network (NN) [5,8,9], Kohonen network [13] where the network is trained considering rotation and occlusion, adaptive resonance theory (ART) network [14] and radial basis function (RBF) as is

* Corresponding author. Fax: +34 918856699.
E-mail addresses: saturnino.maldonado@uah.es (S. Maldonado Bascón), javier.acevedo@uah.es (J. Acevedo Rodríguez), sergio.lafuente@uah.es (S. Lafuente Arroyo), caballer@dsi.uclm.es (A. Fernndez Caballero), francisco.lopez@uah.es (F. López-Ferreras).

1077-3142/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2009.12.002

described in [10]. Alternative classifiers have also been proposed such as a matching based classifier [4], a non-linear correlator in [15], or a normalized correlation-based pattern matching using a traffic sign database in [16,17]. Especially good results have been achieved when the classification method is based on support vector machines (SVM) with a Gaussian kernel [18,11,19].

Nevertheless, the identification of the sign in a TSDRS is technically difficult to realize due to the following problems:

1. Lighting conditions of the scene are changeable according to the time of the day or night and affect the color perception of road signs. So, the illumination level has a great influence on the vectors to identify.
   - For the same scene, the use of different cameras and different settings for the same camera can provoke variations in the image.
   - Although road signs are man-made objects defined by international specifications, there are small variations with respect to pictograms according to manufacturers.
2. The variety of different road signs (classes) to be distinguished is enormous in each country.

As well as dealing with all these issues, a recognition system should also avoid erroneous identification of non-signs, i.e. limit the number of false alarms and finally, even when the purpose is not oriented to real time applications (in our case we are concerned with road maintenance tasks), the computation time should be low. Moreover, the system need to be evaluated with a realistic traffic sign database containing a large enough number of examples. The goal of this work is to combine several pre-processes and improvements on the SVMs to make the traffic sign recognition task less expensive computationally whilst the accuracy is improved.

The paper is organized as follows: Section 2 provides a brief description of the TSDRS on which the optimization recognition process presented in this paper is based. In Section 3, the features of the complete road-sign database used for this work are presented whereas in Section 4, several pre-processing methods are implemented and compared in order to reduce the number of support vectors. Section 5 is focused on the optimization of SVMs and so, the tuning of the hyperparameters associated with the learning machine is described in the sub-Section 5.1, where the optimization of a functional, made by means of Particle Swarm Optimization, allows us to achieve a compromise between the error estimation and the number of support vectors regarding a cost function. This optimization is run independently for each classifier in Section 5.2 and finally, the sub-Section 5.3 proposes an algorithm to reduce the number of support vectors grouping those that are located near enough and can be considered as a unique vector. The final section reports the experimental results that are carried out with our database combining several algorithms. Results demonstrate how the number of support vectors is reduced drastically with the use of the proposed techniques while the accuracy is increased.

2. System overview

An exhaustive description of our TSDRS on which the optimization of this work is based can be found in [11]. The global system allows us to extract signs from a video sequence, and categorize them into specific type categories. The system consists of the following steps:

- Segmentation
- Detection
- Recognition

Although this work is focused in the improvement of the recognition or classification stage, it is important to know how the information is provided to this last stage.

2.1. Segmentation

The segmentation step in traffic sign recognition is a key step in order to get a successful performance. Its purpose is to isolate candidate traffic signs from the background in the scene. Color information is considered in our system to extract candidate objects from the input image by thresholding. The difficulties found in traffic sign segmentation are related to illumination changes and traffic sign deterioration. It is important to point out that in RGB space the high correlation between the three components and the sensitivity to illumination changes makes difficult to find the correct thresholds in an empirical way as it was demonstrated in [20]. Some of the major advantages of the HSI color space are that it has only two color components, Hue and Saturation, and both components are closely related to human perception. Hue represents the dominant color value and Saturation the purity of color with high values given pure colors and low values colors with high mix of white. We think that hue and saturation components of the HSI space are sufficient to isolate traffic signs in a scene working with fixed thresholds under a wide range of illumination condi-

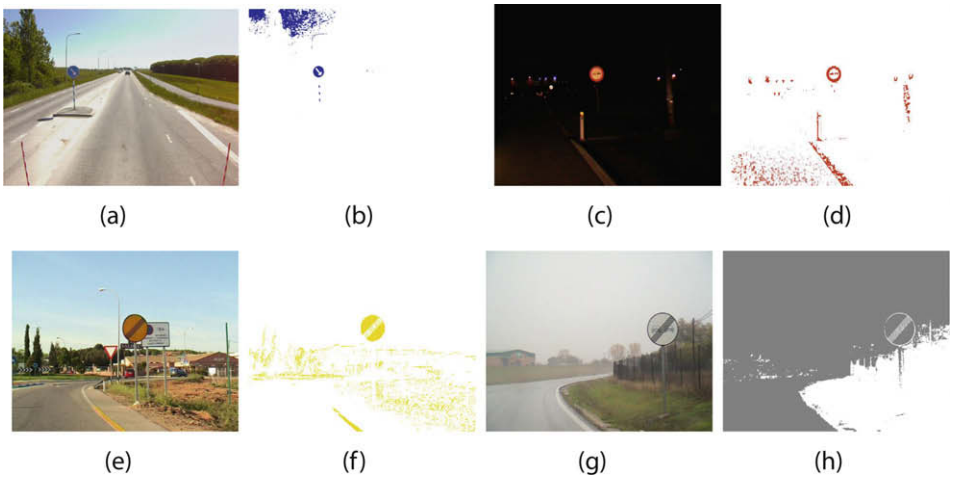

Fig. 1. Examples of segmentation process. (a), (c), (e) and (g) Original images; (b) blue mask; (d) red mask; (f) yellow mask; (h) achromatic mask. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

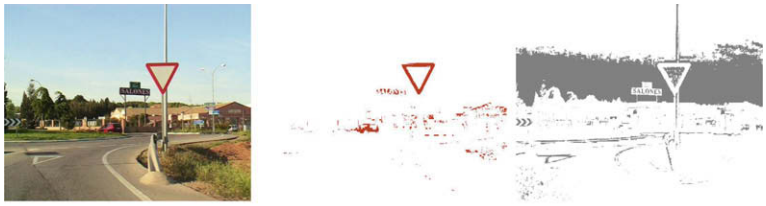

Fig. 2. Example of multiple segmentation. (a) Original image; (b) segmentation mask by red; (c) segmentation mask by achromatic decomposition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

tions. The values of these thresholds have been fixed from the study of hue and saturation histograms. Since hue and saturation components do not contain information to segment white objects, an additional achromatic decomposition similar to the one used in [21], is implemented. Here the ratios among the RGB components are analyzed. In Fig. 1 some segmentation examples are illustrated, including only the interest masks. It is necessary to point out that extracted objects whose sizes in the image are not suitable regarding to a traffic sign are discarded. So, in Fig. 1 the big areas related to the sky and the road are removed.

Most of the common road signs present a red rim and an inner white region. This characteristic leads us to consider the sign as a possible sum of two contributions corresponding to their chromatic and achromatic segmentation masks, where each part is processed independently in the complete system. The advantage of this idea is that a sign can be detected by different colors. In Fig. 2 we can see an example.

2.2. Detection

The detection block can be divided into two sub-blocks. The shape classification sub-block performs the identification of the shape of the blob comparing their signature (defined as the distance from the mass center to the edge of the blob as a function of the angle [22]) with the signature of the theoretical shapes of an equilateral triangle, a square and a circle. Fig. 3 shows the signature of the three references shapes, along with their corresponding shapes. To make the algorithm invariant to object rotations, comparisons are performed using the absolute value of the FFT of the signature signals, instead of the signature itself. Finally, to make the algorithm invariant to object scaling, a normalization of the energy of the signature is also performed. The main advantages of the implemented algorithm are its invariance to object translation, scaling, rotations and a great robustness to camera projection deformation. A complete description of this block can be found in [23]. The output is the blob list returned by the segmentation step updated with its estimated shape.

The other sub-block achieves the localization of the shape. For a triangle and a rectangle, localization means the estimation of the position of its three or four vertices respectively. For instance, in Fig. 4 we can see how these correspondences can be extracted from the previously computed signature. For a circle, it means the estimation of the coordinates of the center, the major and minor axes and the orientation of the major axis of the corresponding ellipse.

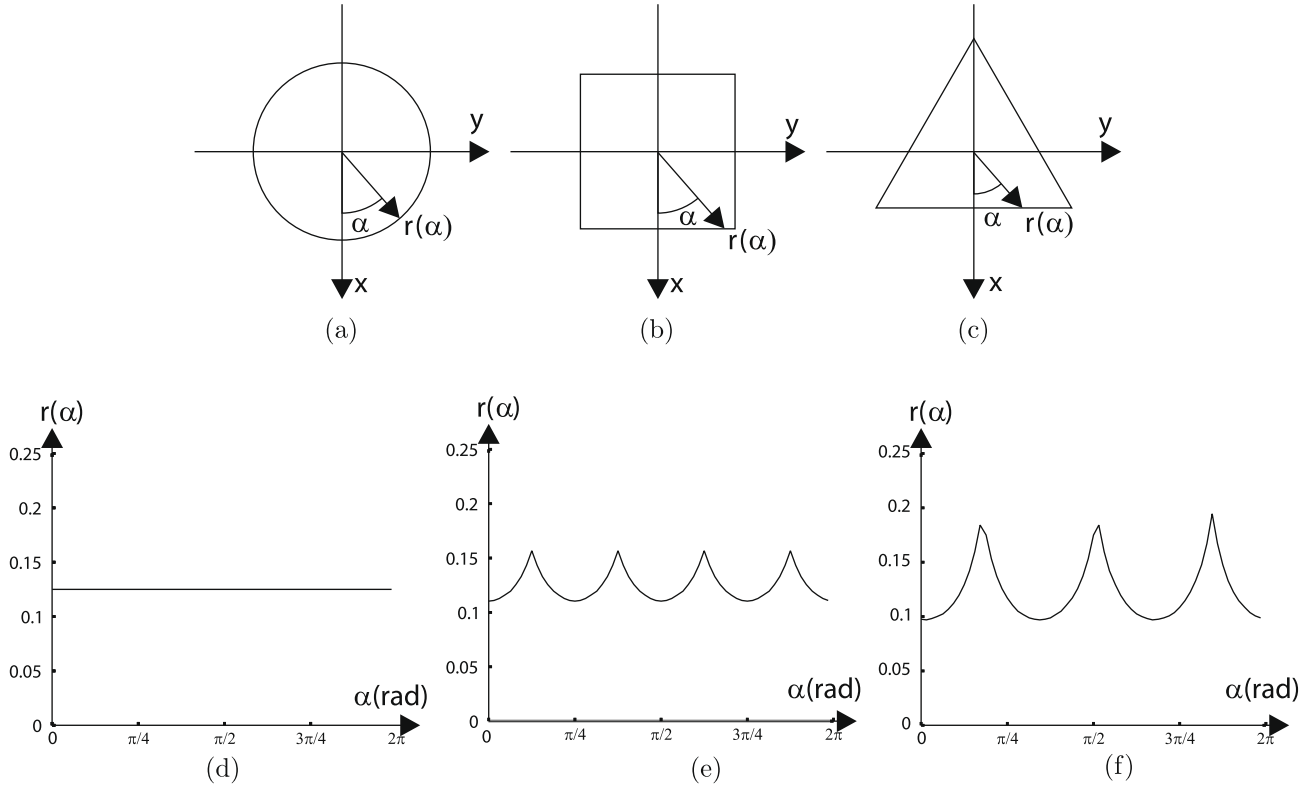

Fig. 3. Shape signature. (a), (b) and (c) Reference shapes; (d), (e) and (f) signatures associated.

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

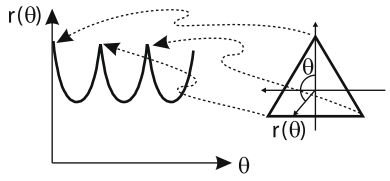

Fig. 4. Triangle correspondences.

(a)

Fig. 5. Positive support vectors for a speed limit sign.

With these points correspondences, a homography can be estimated that allow the system to place the traffic sign in a reference position.

2.3. Recognition

Once a candidate blob has been nominated as a possible traffic sign, the purpose of the recognition stage is to identify the information related to the pictogram of the traffic sign or determines the candidate blob to be noise. In our TSDRS this step is implemented by SVM with Gaussian kernel where the input vector is a normalized size block of 31×31 pixels in gray-scale for each candidate blob. Although a first approach is to use a binarization of the pictograms, the results we obtained in this way were sensitively worst. In real environments, traffic signs are affected by illumination changes, shadows and occlusions, what makes simple thresholding to provoke important confusions at the recognition stage [24]. On the other hand, it is difficult to find an adaptative thresholding, such as the algorithm proposed by Otsu [25], that does not introduce distortion in any pictogram sample due to lack of information respect to the original level of intensity. In addition, the use of binarization in similar pictograms from different classes, for instance speed limit signs, makes the classification system confuse and so, gray-scale is preferred to binary images. In order to reduce the dimensions of feature vectors, only those pixels that cover the area of the traffic sign are computed with the use of a mask, in such a way as is described in [11].

The problem under study is a multi-class one with some noise samples that do not belong to any class. It has to be noted that noise samples come from the segmentation and detection stage and should be identified as noise. In order to simplify the difficulties of the recognition problem, both the training and testing are done according to the color and shape of each candidate region. Thus, every object is only compared with those signs that have the same color and geometric properties as the blob to be identified. The extension of the SVMs learning method in this case is done following a one-against-all strategy [26] and so, the number of classifiers M needed is equal to the number of classes that belong to the case considered, such as red circular or red triangular. This means that in the one-against-all strategy, we have not built a binary classifier to separate noise from the rest, but noise samples are included in the group named rest in every binary classifier. When none of the classifiers gives a positive output, the sign under study is considered to be noise. When two or more classifiers give positive outputs, the one with highest output is taken. Each one of the different M one-vs.-all classifiers are implemented in our initial system using a radial basis function (RBF) kernel (1), since other kernels, such as polynomial or sigmoidal ones, gave worse results.

$$K(\boldsymbol{x},\boldsymbol{y}) = e^{-\gamma \sum_{i=1}^{n}(x_i - y_i)^2}. \quad (1)$$

We must point out that only some pattern vectors of the whole set define the decision hyperplane. These pattern vectors are known as support vectors. As an example, Fig. 5 shows the positive support vectors that define the binary decision region for the speed limit traffic sign of 100 km/h.

Due to the size normalization of each blob, the method is invariant to scale changes. Results obtained for a video sequence are illustrated in Fig. 6, where all identified objects are drawn over the image with their respective geometrical shape. As is explained above, a road sign can be segmented twice due to two contributions: chromatic and achromatic.

3. Road-sign database

In the creation of a considerable traffic sign database, the most important aspects to consider are, firstly, the possible variations regarding lighting conditions and different pictograms for the same class as we can see in Fig. 5 and, secondly, the high number of different classes for each segmentation and possible shapes associated. However, although there are numerous works related to road sign identification, we have not been able to find any standard traffic sign set.

For this purpose, the Recognition and Multi-sensorial analysis group (GRAM) at the Universidad de Alcalá have collected a complete database of Spanish traffic signs, which is summarized in Table 1 for some of the different sets. As we can see, the number of collected patterns for blue signs is lower than for red signs since the blue ones are less frequent in roads. All the samples have been extracted from images acquired by different video-cameras under variable lighting conditions. The stored patterns are gray level and have 31×31 pixels of size with homogeneous background

(a) (b) (c)

Fig. 6. Results of a video sequence S1 at recognition output.

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

Table 1
Description of the different data sets used.

Data set	Number of samples	Number of classes	Number of noise samples
Red triangular	12177	46	3010
Red circular	17632	63	3676
Blue circular	2628	9	521
Blue squared	3277	35	865

for no-interest pixels, as is mentioned above. Thus, the number of significant components is 961 for rectangular signs while in circular signs it is reduced to 709 and for triangular signs to 511.

Although results shown in this work have been obtained with the mentioned database, the training set is easily configurable and a new set can be defined for specific traffic signs from other countries. In fact, the system has been applied to Canadian and European signs in the same way that in the case of Spanish signs. Results we have obtained are similar for all cases.

4. Image enhancement

One of the problems to overcome in traffic sign recognition is the variation of illumination conditions we can find in natural outdoor environments, as has been mentioned. On the other hand, the computational cost in SVMs classification is related directly to the number of support vectors, which increases as the number of classes and the disparity of the samples do in the training set. For this reason, we try to improve the success ratio and reduce, in some cases, the number of support vectors according to the fact that, as we can observe in Fig. 5, many support vectors are very similar except in the illumination level.

Thus, we search for several methods to process images, which give us images more suitable than the original ones in order to get a more homogenous set for this application.

4.1. Gray level normalization

Since our dataset present different variations of illumination, it seems obvious that one alternative to reduce this effect consists of a normalization gray level average. In this way, all image histograms are centered over a fixed gray level value K. In Fig. 8 the results are shown after this transformation is being applied with $K = 120$ for the original set shown in Fig. 7.

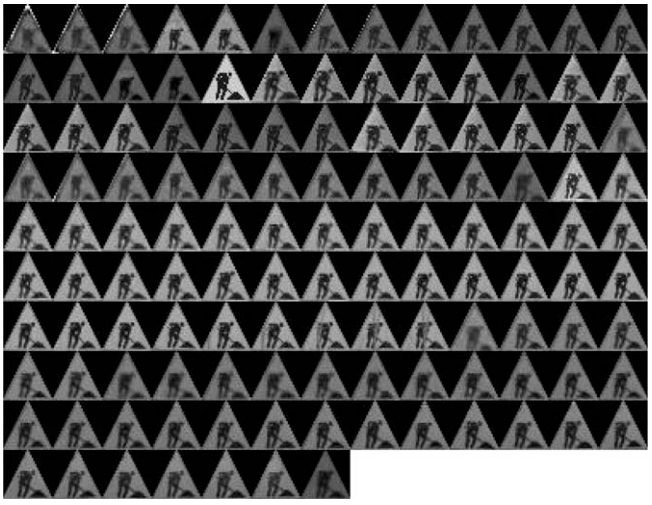

Fig. 7. Examples of 'road construction' traffic sign set.

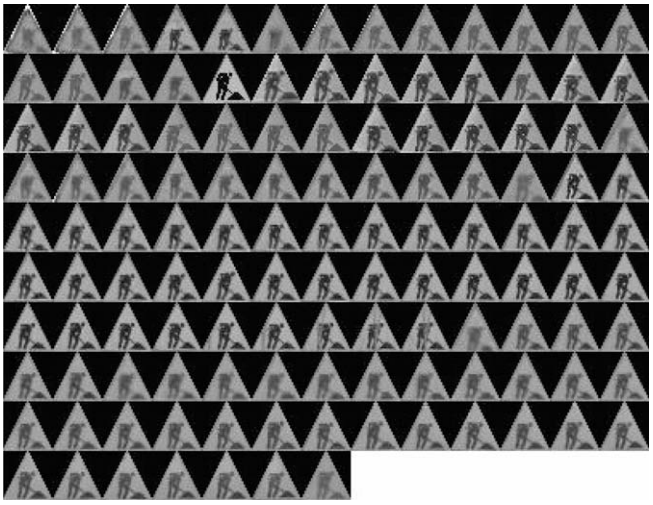

Fig. 8. Results obtained over examples of 'road construction' traffic sign set after applying a level-gray normalization.

4.2. Contrast stretching

One of the alternatives to reduce the lighting variations is to use piecewise linear functions through a contrast-stretching transformation. Low-contrast images can result from poor illumination, lack of dynamic range in the imaging sensor, or even wrong setting during image acquisition.

Fig. 9 shows a transformation used for contrast stretching. The locations of points (r_1, s_1) and (r_2, s_2) control the shape of the transformation function. If $r_1 = s_1$ and $r_2 = s_2$, the transformation is a linear function that produces no changes in gray levels. In this case, we have applied this technique setting $(r_1, s_1) = (r_{min}, 0)$ and $(r_2, s_2) = (r_{max}, L-1)$, where L is the number of gray levels, i.e. 256. The values of the parameters r_{min} and r_{max} are computed in order to achieve that at least P pixels have gray level lower than r_{min} and other P pixels have higher than r_{max}. Fig. 10 illustrates the results obtained over the set of Fig. 7 using the described method when the value of the parameter P has been fixed at 40.

4.3. Equalization

This method is well known and usually increases the local contrast, especially when the usable data of the image is represented

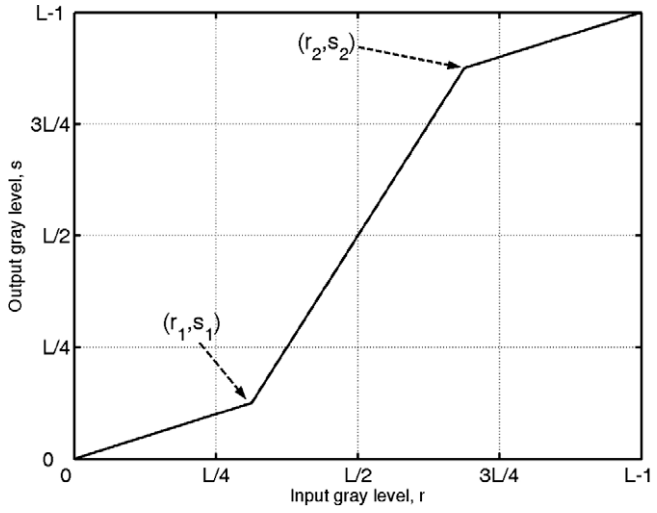

Fig. 9. Contrast-stretching transformation.

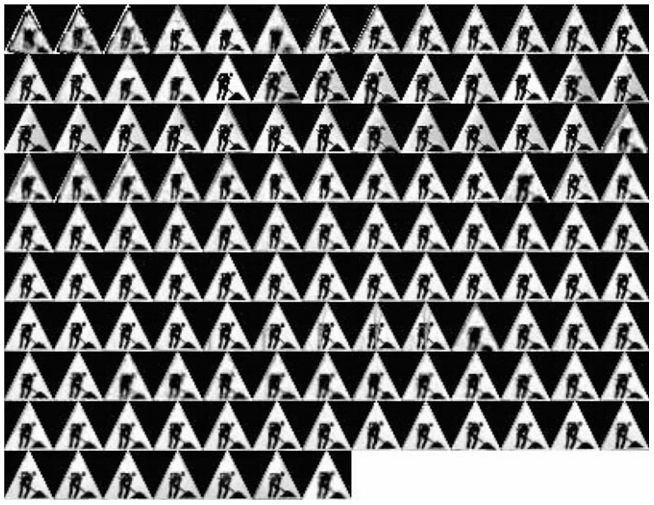

Fig. 10. Results obtained over examples of 'road construction' traffic sign set after applying contrast stretching for $P = 40$.

by close contrast values. Through this adjustment, the intensities can be better distributed on the histogram. This allows for areas of lower local contrast to gain a higher contrast without affecting the global contrast. Histogram equalization accomplishes this by effectively spreading out the most frequent intensity values.

5. Improving support vector machines

The main objectives of the work require the final recognition stage to be based on a method with a high accuracy but at the same time, the memory used and the number of operations in the test phase have to be minimized. As it has been mentioned, the use of SVMs in traffic sign recognition gives good results and this method is considered a sparse learning method [27]. In this work the software described in [28] is used to train the system and to obtain support vectors, while L1 SVMs were used due to the fact that with this last formulation the number of support vectors is more reduced than with the L2 SVM method [29].

The target of this part of the work is to reduce the number of support vectors and thereby to reduce memory requirements and the number of operations needed in the test phase, with the highest possible accuracy.

5.1. Hyperparameter optimization by means of particle swarm optimization

One of the most difficult points in classification is to tune the parameters associated with the learning machine. In our case, when working with SVMs, the choice of the kernel is going to have a great influence on the success of the classifier. When the M classifiers are built, most of the one-against-all problems are not linearly separable and, as it has been mentioned, a non-linear kernel such as the one proposed in Eq. (1) has to be applied.

Support vector machines training requires setting a priori a regularization constant C [26]. When the radial basis function kernel is used, in addition to the C parameter, the γ parameter has to be tuned. In (Fig. 11) it can be appreciated how the estimation of the error varies with these two parameters for a given classification experiment. This error estimation can be done using different methods such as the span estimator described by Vapnik and Chapelle [30], the $\xi\alpha$ estimator introduced by Joachims [31] or the popular leave-one-out, but in our case there are enough samples to estimate this error with a separate test set, whereas final results are calculated with a different validation set, to avoid overfitting of the system. Thus, the estimated error calculated over a test set with n test samples is given by Eq. (2) where $f_{C,\gamma}(\boldsymbol{x}_i)$ is the output of the classifier trained with $\boldsymbol{\theta} = [C, \gamma]$ as common hyperparameters for all the binary classifiers and y_i is the real class associated with the input vector $\boldsymbol{x}_i$.

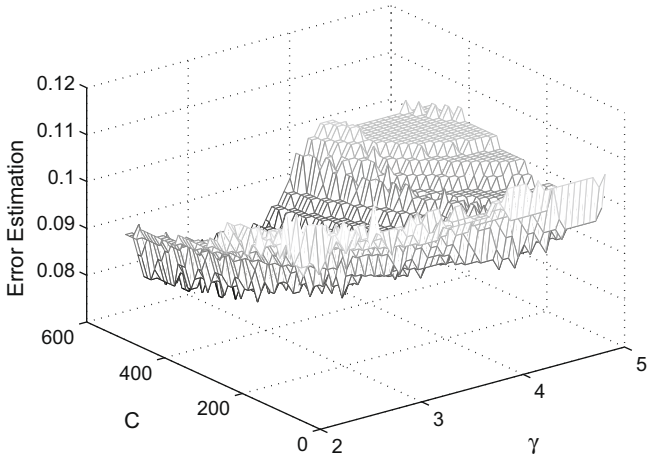

Fig. 11. An example of the error with different values for C and γ.

$$\hat{\epsilon}_{C,\gamma} = \frac{1}{n}\sum_{i=1}^{n}(1 - \delta(y_i, f_{C,\gamma}(\boldsymbol{x}_i))).$$
$$\delta(y_i, f_{C,\gamma}(\boldsymbol{x}_i)) = \begin{cases} 1, & y_i = f_{C,\gamma}(\boldsymbol{x}_i) \\ 0, & y_i \neq f_{C,\gamma}(\boldsymbol{x}_i) \end{cases} \quad (2)$$

Besides optimizing the hyperparameters to achieve a minimum in the error estimation, our target is to reduce the number of support vectors and therefore the number of operations needed in the test phase. Instead of searching for optimal hyperparameters to optimize Eq. (2) our proposal is to minimize the functional $L(C, \gamma)$ described as

$$L(C,\gamma) = \begin{cases} \hat{\epsilon}_{C,\gamma} + \frac{N^T_{SV(C,\gamma)}}{ln}, & \hat{\epsilon}_{C,\gamma} \geqslant \lambda \\ \lambda + \frac{N^T_{SV(C,\gamma)}}{ln}, & \hat{\epsilon}_{C,\gamma} < \lambda \end{cases} \quad (3)$$

where l is the number of samples used to train the SVMs, λ is a given error threshold below its value, we focus only on the reduction of the number of support vectors and $N^T_{SV(C,\gamma)}$ is the number of all support vectors among all the binary classifiers. This functional selects the hyperparameter that minimizes error if its value is above the λ parameter. If two hyperparameter combinations produce the same error, then the second term of the expression ensures that the one with less support vectors is selected. Although other strategies are also possible, with the expression of this second term it is guaranteed that above the λ value the functional only minimizes the number of support vectors in the event of a tie. When the error is below the λ value, the functional selects the combination with less support vectors. Now that the functional to be minimized is defined, the problem is how to find such hyperparameters. Due to the fact that large regions in the search space give the same estimated error, this search is usually done in several works by means of a grid search [32], but this method is not adequate to optimize the functional $L(C, \gamma)$ because the grid step should be small enough to take into account variations in the number of support vectors and so, the number of training processes to be evaluated would be huge. Instead of this, our proposal is to use a statistical local search (SLS) method to find the optimal parameters.

Particle swarm optimization (PSO) [33] is a recent method for function minimization and it is inspired by the emergent motion

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

of a flock of birds searching for food. This algorithm has been proposed in preliminary works by our group to tune SVMs hyperparameters [34]. Like in other SLS, the search for the optimum is an iterative process that is based on guided random decisions taken by m particles searching the space at the same time. Each particle i has an initial position $\boldsymbol{\theta}_i$ that is a vector with a possible solution to the problem. In our case, the components of the vector are the C and γ parameters if the kernel is RBF. Each position is evaluated with the objective function described in Eq. (3) and the particles update their position according to (4) where $\boldsymbol{v}_i(t+1)$ is the new velocity of particle i, $\phi(t)$ is the linear inertia function as it is described in [35], $\boldsymbol{p}^i_{best}$ is the best position achieved by the particle i, $\boldsymbol{g}_{best}$ is the best position achieved by any of the particles, $c_{1,2}$ are coefficients, fixed at a value of 2 in this work, related to the strength of attraction to the $\boldsymbol{p}_{best}$ and $\boldsymbol{g}_{best}$ position respectively and $r_{1,2} \in [0,1]$ are random numbers generated with a uniform random distribution.

$$\begin{aligned} \boldsymbol{v}_i(t+1) &= \phi(t)\boldsymbol{v}_i(t) + c_1 r_1(\boldsymbol{p}^i_{best} - \boldsymbol{\theta}_i) + c_2 r_2(\boldsymbol{g}_{best} - \boldsymbol{\theta}_i) \\ \boldsymbol{\theta}_i(t+1) &= \boldsymbol{\theta}_i(t) + v_i(t+1). \end{aligned} \tag{4}$$

The search for each particle is done using its past information and the neighborhood one. This fact makes the particles fly to a minima position but they can escape if it is a local minima.

5.2. Optimizing binary classifiers

In the previous section, a method to optimize the hyperparameters was exposed achieving a minimum error and reducing the number of support vectors where the same kernel and same hyperparameters were used for all the binary classifiers. It is clear that there are some kinds of traffic signs that are quite different from the rest and it would be enough to separate them from the rest with a linear kernel. When this happens, this kernel should be selected because it is the one with fewer operations required and only one equivalent vector support has to be stored in memory. On the other hand, the variance of each kind of traffic sign does not have to be so similar, and therefore, a good strategy to improve the accuracy and the number of support vectors is to search for the optimal hyperparameters for each binary classifier. In this part of the work, we propose to use the following steps to design each binary classifier.

1. *Calculate the estimated error with a linear kernel.* As it has been mentioned a SVM trained with the linear kernel implies only to calculate a dot product in the test phase, so the first step is to estimate the best error found with a linear kernel and store it as $\hat{\epsilon}_l$. Note that a SVM trained with a linear kernel only requires to tune the C parameter and this can be done by a sweep of several values of this parameter. In our case, we have selected a sweep from $C = 10$ to $C = 1000$ with 20 steps. No better results were found if the range or number of steps were increased.
2. *Estimate the error with the RBF kernel.* If $\hat{\epsilon}_l \leqslant \lambda$ the linear kernel ensures to be the best option to minimize Eq. (3) and so this second step can be avoided. Otherwise search with PSO for the hyperparameters $\boldsymbol{\theta} = [C, \gamma]$ that minimizes Eq. (3) and store the error achieved with these hyperparameters as $\hat{\epsilon}_{RBF}$.
3. *Select the appropriated kernel.* if $\hat{\epsilon}_l \leqslant \hat{\epsilon}_{RBF}$ select the linear kernel with the corresponding C value. Otherwise, select the RBF kernel with the hyperparameters $\boldsymbol{\theta} = [C, \gamma]$ found in the previous step.

However, there is an objection to this focus. When all the binary classifiers were built with the same kernel, including the hyperparameters, the implicit transformation of the kernel to a Hilbert space was the same for each classifier, which means that it was possible to compare the outputs, but in this case this comparison cannot be done. To solve this problem, it is possible to add a stage for each classifier that maps the output into a probability as it is described by Platt [36]. In such a way, the classifier with the highest output will be considered if it is above a given threshold. If all the outputs are below the threshold, the sign under study is considered noise.

5.3. Grouping of support vectors

Once the optimal hyperparameters have been found, it is still interesting to reduce the requirements of memory and number of operations needed in the test phase. Support vectors are samples from the training set that lie on the margin border ($0 < \alpha_i < C$), are inside the margin zone or even misclassified training samples ($\alpha_i = C$). It can be appreciated in real data sets how it is highly probable that support vectors not appear as isolated samples, but there are groups of samples with small euclidean distance that are selected as support vectors. The proposal of this task is to concentrate into a unique support vector those samples that are close enough. This algorithm will be applied to each of the binary classifiers in an independent way of the strategy chosen to optimize the hyperparameters.

The algorithm to compute this reduction is as following:

1. Create two matrices, $\boldsymbol{D}^+$ and $\boldsymbol{D}^-$ whose elements are calculated as follows:

$$\begin{aligned} D^+(i,j)_{i \neq j} &= \|\boldsymbol{sv}^+_i - \boldsymbol{sv}^+_j\| \\ D^-(i,j)_{i \neq j} &= \|\boldsymbol{sv}^-_i \boldsymbol{sv}^-_j\| \\ D^+(i,i) &= D^-(i,i) = \beta \end{aligned} \tag{5}$$

where $\boldsymbol{sv}^+_i$ is the ith positive support vector and $\boldsymbol{sv}^-_i$ is the ith negative support vector. The last part of Eq. (5) is added to avoid considering the null distance between a support vector and itself, so in the created matrices it is assigned a value β that will ensure that these distances are not taken into account.
2. Search for the minimum distance D^+_{min} and check if $D^+_{min}(i,j) < \beta$ where β is a given threshold. If this condition is not fulfilled, the grouping of positive support vectors has finished. The same applies to negative support vectors if $D^-_{min}(i,j) < \beta$
3. Remove one of the support vectors (i.e. $\boldsymbol{sv}_j$) that has the minimum distance. The reason to select one of them and not to create a pseudo-vector is to allow that the same vector can be found in another classifier, saving memory.
4. Rebuild the matrix $\boldsymbol{D}$ where the reduction has been done, or both matrixes if there was reduction in both cases. The new matrix becomes $\boldsymbol{D}^{\setminus i}$, that is the matrix $\boldsymbol{D}$ without row and column i.
5. Repeat the process from step 2 until it is not possible to group support vectors.
6. Retrain the SVM only with the support vectors not discarded.

It has to be noted that this grouping of support vectors only makes sense in those binary classifiers that are built with a nonlinear kernel, because as it was mentioned for linear kernels we can store the hyperplane instead of the support vectors. It is also important to note that we are searching for support vectors that are near in a euclidean sense. However, one can deduce that it should be more correct to search for neighborhood in the transformed Hilbert space implicit in the kernel mapping. Taking into account that we can calculate the distance in the transformed space [37] as:

$$\|\phi(\boldsymbol{sv}_i) - \phi(\boldsymbol{sv}_j)\|^2 = K(SV_i, SV_i) + K(SV_j, SV_j) - 2K(SV_i, SV_j) \tag{6}$$

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

When using RBF kernels the previous equation becomes

$$\|\phi(\boldsymbol{sv}_i) - \phi(\boldsymbol{sv}_j)\|^2 = 2 - 2e^{-\gamma\|(\boldsymbol{sv}_i)-(\boldsymbol{sv}_j)\|^2} \quad (7)$$

and so, search for the minimum $\|\phi(\boldsymbol{sv}_i) - \phi(\boldsymbol{sv}_j)\|^2$ implies a search for the minimum euclidean distance in the input space.

The main parameter of the proposed algorithm is β, where a low value of this parameter means that only very close support vectors are grouped and decision function does not change very much, what implies a very similar performance than before applying the method. A very high value of the β parameter means that most of the support vectors are grouped, changing the decision function and the performance of the SVM drops.

6. Results and discussion

In this section the different methods and ideas exposed are tested to verify their capabilities to increase the accuracy and to reduce the number of support vectors. All the datasets used are divided in three sets, a training one, a test one to tune the necessary parameters and a validation one used to present the results. The division was done randomly but with a supervision to keep the same probabilities of each class of the whole dataset. In all the datasets there are noise signs from the segmentation stage that are included in each binary classifier with the rest of the samples that are not the class under study. Datasets used are described in Table 1.

6.1. Comparing pre-processing methods and hyperparameter selection

In this section we would like to compare the exposed pre-processing methods and the strategies for the multiclass problem in two ways. On the one hand, we would like to check the improvements on the total accuracy and the rejection of noise signs, but on the other hand, it is also the target of this part of the work to verify the reduction in the number of support vectors and so, the number of operations needed for every new traffic sign to be classified and the amount of memory necessary to store the learning machines. So, for each dataset the following pre-processing methods were applied:

- Gray level normalization.
- Histogram equalization.
- Contrast stretching. In this work we have used $r_{min} = r_{max} = 10$.

These pre-processing methods produce then three different data sets plus the one without pre-processing for each of the ones described in Table 1. Every dataset obtained is trained, tuned and validated with three different strategies:

Fixed hyperparameters: In this case, all the binary classifiers are trained with RBF kernels and the same hyperparameters for all of them. The values of these hyperparameters are described in [11].

Optimized common hyperparameters: As in the previous case, all the binary classifiers are trained with RBF kernels and the same hyperparameters for all of them but in this case the values of the hyperparameters are optimized for each dataset using PSO to minimize functional (3). PSO was executed with 10 particles and a maximum of twelve generations. The λ parameter of the functional (3) is set to a value of $\lambda = 0.02$ as total error allowed. Although this tolerance parameter can be fixed to any other value, the one selected fulfils the requirements needed for our system.

Optimized per classifier: In this case, each binary classifier has its own hyperparameters and the kernel is chosen between the linear one or the RBF one, as it has been described. PSO is applied to each binary classifier when the RBF kernel has to be tested, giving optimized hyperparameters for each classifier. In this case the PSO algorithm is executed with eight particles and a maximum of seven generations, whereas the λ parameter was reduced to $\lambda = 0.002$. Note that in this second case, the selected value refers to the allowed error for each binary classifier of a one versus all strategy, so the value must be reduced in order to get similar overall accuracy and in our test the selected value of achieves a total tolerance similar to the previous case.

Results, accuracy and number of support vectors, are shown in the following Tables 2–5 for red triangular, red circular, blue circular and blue rectangular traffic signs. Accuracy for a given set of hyperparameters ($\hat{\rho}_\theta$) is defined as:

$$\hat{\rho}_\theta = 1 - \hat{\epsilon}_\theta \quad (8)$$

being ϵ_θ the estimated error defined in Eq. (2). The improvement of the pre-processing methods can be seen in each column, whereas results for the different training strategies are appreciated in each row.

In these tables, it can be appreciated how the best accuracy results are obtained with histogram equalization and contrast stretching pre-processing methods, but when the common hyperparameters strategy is used the number of support vectors even increase with respect to the first strategy where no optimization was done. This behavior is quite logical as the optimization process is focused on accuracy where the error is above the λ parameter. Only in those cases where the accuracy is good enough, the number of support vector is optimized by the proposed method. The strategy based on optimizing each classifier shows similar accuracy results to the common hyperparameter one, but the number of support vectors fall down in a sensitive way, which is explained due to the fact that the optimization in support vector number is easier done in some binary classifiers, even using linear kernels, whereas more support vectors can be spent in those binary classifiers that require more complex frontiers. Although the reduction of the number of support vectors in red triangular and circular signs is quite important, even above a 50% in some cases, a much significant improvement can be appreciated in blue signs. The main reason for this is the number of linear binary classifiers that can be built within these kinds of signs. With this optimization strategy, histogram equalization and contrast stretching show good results in accuracy and the number of support vectors, but the former is selected due to the reduction in red signs, that have greater presence in roads, and the number of operations needed in the pre-processing stage.

It has to be highlighted that the proposed method to optimize the hyperparameters implies several hours to get trained the whole database. Nevertheless, what is really important in TSDRS is to reduce time once the system is trained, which is achieved with the reduction of the number of support vectors. Average classification time strongly depends on the system employed and the amount of information is considered. This average time, with the proposed reduction scheme, is about one second per image if all the signs are considered.

Finally, to probe the statistical significance of the achieved improvements the conclusions obtained in [38] are followed. Given two different classifiers with estimated accuracies $\hat{\rho}_1$ and $\hat{\rho}_2$, with $\hat{\rho}_2 > \hat{\rho}_1$, we would like to probe if the observed difference $\triangle\hat{\rho} = \hat{\rho}_2 - \hat{\rho}_1$ is statistically significant. From the mentioned work and assuming i.i.d errors, it can be ensured with a risk α that $\triangle\hat{\rho}$ is statistically significance if the following criteria is fulfilled:

$$\triangle\hat{\rho}^2 = \frac{4z_\alpha^2\bar{\rho}}{n} \quad (9)$$

Table 2
Accuracy and number of support vectors for the red triangular traffic signs dataset.

	Fixed hyperparam.		Opt. common hyperparam.		Optimized per classifier	
	Acc	Nsv	Acc	Nsv	Acc	Nsv
Without pre-processing	0.928	5263	0.931	5714	0.925	4715
Gray level normalization	0.935	4271	0.940	4695	0.928	3804
Histogram equalization	0.952	7057	0.935	5853	0.926	4515
Contrast stretching	0.963	6316	0.962	6433	0.956	2901

Table 3
Accuracy and number of support vectors for the red circular traffic signs dataset.

	Fixed hyperparam.		Opt. common hyperparam.		Optimized per classifier	
	Acc	Nsv	Acc	Nsv	Acc	Nsv
Without pre-processing	0.931	6884	0.940	7505	0.936	5995
Gray level normalization	0.927	6329	0.945	6717	0.939	5297
Histogram equalization	0.953	8249	0.954	10045	0.952	4519
Contrast stretching	0.960	6961	0.960	7305	0.956	3771

Table 4
Accuracy and number of support vectors for the blue circular traffic signs dataset.

	Fixed hyperparam.		Opt. common hyperparam.		Optimized per classifier	
	Acc	Nsv	Acc	Nsv	Acc	Nsv
Without pre-processing	0.977	1563	0.977	1057	0.975	257
Gray level normalization	0.977	1491	0.978	1072	0.972	180
Histogram equalization	0.979	2042	0.984	1591	0.986	132
Contrast stretching	0.980	3095	0.983	1289	0.983	225

Table 5
Accuracy and number of support vectors for the blue rectangular traffic signs dataset.

	Fixed hyperparam.		Opt. common hyperparam.		Optimized per classifier	
	Acc	Nsv	Acc	Nsv	Acc	Nsv
Without pre-processing	0.952	3818	0.961	2269	0.958	527
Gray level normalization	0.959	3590	0.960	3938	0.957	673
Histogram equalization	0.950	6536	0.983	1289	0.963	96
Contrast stretching	0.959	8060	0.963	2785	0.962	608

where n is the number of samples of the dataset used to estimate $\hat{\rho}_1$ and $\hat{\rho}_2, z_\alpha$ can be determined from tables of the normal law and $\bar{\rho}$ is the true mean obtained for the considered classifiers. Assuming for this last value $\bar{\rho} = 0.98$ and a third of the number of samples used to estimate the accuracy the minimum statistically significant different can be obtained. Table 6 shows, with a risk $\alpha = 0.05$, these minimum differences for the studied datasets and it can be appreciated how the improvements achieved can be considered statistically significant.

In order to show how the system performs on a typical route, a video sequence is included in [39] where it can be appreciated how in a large video sequence the system detects all the traffic signs. The sequence also gives a clear idea about the behavior of the recognition stage. Once the traffic sign has been detected, there are some frames where the traffic sign is considered to be noise, what is reflected in the sequence as a flickering in the traffic sign icon. However, it also can be shown how the number of false positives is null in this case.

6.2. Grouping support vectors

Once we have studied the improvements achieved by the preprocessing stage plus an adequate hyperparameter optimization strategy in accuracy and the number of support vectors sense, in this section the results when the exposed support vector grouping method is used are presented. From here on, the pre-processing method used is contrast stretching and the optimization is done per classifier. In Figs. 12 and 13, the accuracy and the number of support vectors against the distance threshold β for circular and triangular red signs are shown. As it was mentioned there is a commitment in the reduction of support vectors and the performance of the SVM, controlled by the β*parameter*. It can be appreciated how the accuracy nearly remains fixed for distance thresholds that are not high, whereas the number of support vectors falls down quickly. For higher values of β, the accuracy begins to fall until it reaches unacceptable values. So, we can achieve a quite similar accuracy with a reduction of 25–35% of the support vectors number.

As an example, in the case of red circular signs, showed in Fig. 12 before the grouping is applied, there are 3700 support vectors with an accuracy of 0.955. After the method is applied, the

Table 6
Minimum differences with statistical significance ($\alpha = 0.05$).

Data set	Number of test samples	$\triangle\hat{\rho}_{min}$
Red triangular	4059	0.0513
Red circular	5877	0.0426
Blue circular	876	0.1103
Blue squared	1093	0.0988

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002

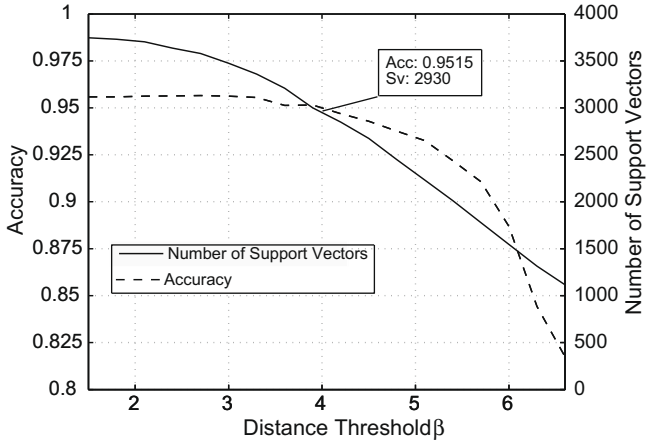

Fig. 12. Groping support vectors in red circular traffic signals.

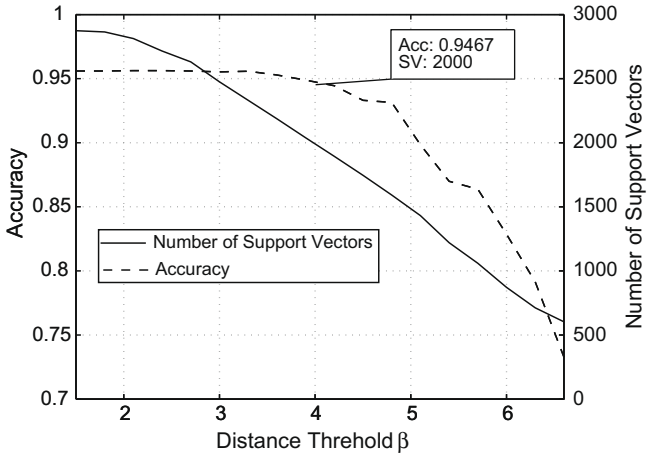

Fig. 13. Groping support vectors in red triangular traffic signals.

point highlighted in the figure shows a number of support vectors of 2930 with an accuracy of 0.951. So, it can be appreciated how a good choice of the β parameter results in important reduction in the number of support vectors whilst the performance of the SVM keeps near the same.

7. Conclusion

In this work we have presented several methods to improve the accuracy of an automatic traffic sign detection and recognition system and, at the same time, to reduce the number of support vectors needed and thus, reduce the requirements of memory and time to test new samples. Contrast stretching has been demonstrated to be a good technique to increase the accuracy of the system and partially avoids illumination changes in the signs to be detected. Histogram equalization also improves the accuracy, but the number of operations to pre-process the image is greater than in contrast stretching.

Once the signs have been pre-processed, some techniques to improve accuracy and to reduce the number of support vectors are introduced. The first of these techniques is to tune the hyperparameters, that play an important role in the performance of the SVMs, by means of particle swarm optimization to achieve a faster search than testing by grid and allow the tuning of the hyperparameters with a functional that reduces the error and the number of support vectors. This strategy is extended to optimize each binary classifier with its own hyperparameters and also training with linear kernels that are good enough for some binary classifiers. Finally, an extra reduction of the number of support vectors can be achieved grouping those support vectors that are near enough. This reduction can be done keeping the accuracy in quite similar levels as the one achieved with the entire set of support vectors. Results presented indicate that accuracy is increased 3% whereas the number of support vectors is reduced to half. The proposed methods are not only applicable to the traffic sign recognition problem but could be a solution to a large variety of problems with similar requirements than the traffic sign recognition problem.

Acknowledgment

This work was supported by Ministerio de Educación y Ciencia de España under Projects TEC2004/03511/TCM and TEC2008-0277/TEC. Authors also acknowledge financial support given by the Comunidad de Madrid Projects CCG06-UAH/TIC0695 and CCG07-UAH/TIC-1740.

References

[1] F.-Y. Wang, P. Mirchandany, Z. Wang, The VISTA project and its applications, IEEE Intelligent Systems 17 (6) (2002) 72–75.
[2] P. Siegmann, R. López, P. Gil, S. Lafuente, S. Maldonado, Fundamentals in luminance and retroreflectivity measurements of vertical traffic signs using a color digital camera, IEEE Transactions on Instrumentation and Measurement 57 (3) (2008) 607–615.
[3] R. Vicen Bueno, R. Gil-Pita, M. Rosa-Zurera, M. Utrilla-Manso, F.López-Ferreras, Multilayer perceptrons applied to traffic sign recognition tasks, in: Proceedings of the 8th International Work-Conference on Artificial Neural Networks, IWANN, Vilanova i la Geltru, Barcelona, Spain, 2005, pp. 865–872.
[4] T. Hibi, Vision based extraction and recognition of road sign region from natural color image, by using HSL and coordinates transformation, in: Proceedings of the 29th International Symposium Automotive Technology and Automation, Florence, Italy, 1996.
[5] D. Kang, N. Griswold, N. Kehtarnavaz, An invariant traffic sign recognition system based on sequential color processing and geometrical transformation, in: Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation, Dallas, TX, 1994, pp. 88–93.
[6] S.H. Hsu, C.L. Huang, Road sign detection and recognition using matching pursuit method, Image and Vision Computing 19 (2001) 119–129.
[7] P. Douville, Real-time classification of traffic signs, Real-Time Imaging 6 (3) (2000) 185–193.
[8] Y. Aoyagi, T. Asakura, A study on traffic sign recognition in scene image using genetic algorithms and neural networks, in: Proceedings of the 22nd IEEE International Conference on Industrial Electronics, Control and Instrumentation, Taipeh, Taiwan, vol. 3, 1996, pp. 1838–1843.
[9] A. de la Escalera, L.E. Moreno, E.A. Puente, M.A. Salichs, Neural traffic sign recognition for autonomous vehicles, in: IECON'94. 20th International Conference on Industrial, Electronics, Control and Instrumentation, IEEE, New York, USA, vol. 2, 1994, pp. 841–846.
[10] U. Krebel, F. Lindner, C. Wohler, A. Linz, Hypothesis verification based on classification at unequal error rates, in: Proceedings of the IEEE Artificial Neural Networks, 1999, pp. 874–879.
[11] S. Maldonado, S. Lafuente, P. Gil, H. Gomez, F. Lopez, Road-sign detection and recognition based on support vector machines, IEEE Transactions on Intelligent Transportation Systems 8 (2) (2007) 264–278.
[12] P. Paclick, J. Novovicova, R. Duin, Building road-sign classifiers using a trainable similarity measure, IEEE Transactions on Intelligent Transportation Systems 7 (3) (2006) 309–321.
[13] R. Luo, H. Potlapalli, D. Hislop, Neural network based landmark recognition for robot navigation, in: Proceedings of International Conference on Industrial Electronics, Control, Instrumentation, and Automation, Power Electronics and Motion Control, 1992.
[14] A. de la Escalera, J.M. Armingol, M. Mata, Traffic sign recognition and analysis for intelligent vehicles, Image and Vision Computing 21 (2003) 247–258.
[15] E. Perez, B. Javidi, Nonlinear distortion-tolerant filters for detection of road signs in background noise, IEEE Transactions on Vehicular Technology 51 (3) (2002) 567–576.
[16] J. Miura, T. Kanda, Y. Shirai, An active vision system for real-time traffic sign recognition, in: Proceedings of the IEEE Intelligent Transportation Systems, 2000, pp. 52–57.
[17] A. de la Escalera, J.M. Armingol, J.M. Pastor, F.J. Rodríguez, Visual sign information extraction and identification by deformable models for intelligent vehicles, IEEE Transactions on Intelligent Transportation Systems 15 (2) (2004) 57–68.

[18] E. Frfas-Martfnez, A. Snchez, J. VTlez, Support vector machines versus multilayer perceptrons for efficient off-line signature recognition, Engineering Applications of Artificial Intelligence 19 (6) (2006) 693–704.
[19] B. Cyganek, Circular road signs recognition with soft classifiers, Integrated Computer-Aided Engineering 14 (4) (2007) 323–343.
[20] H. Kamada, S. Naoi, T. Gotoh, A compact navigation system using image processing and fuzzy control, in: Proceedings of the Southeastcon, New Orleans, vol. 1, 1990, pp. 337–342.
[21] H. Liu, D. Liu, J. Xin, Real-time recognition of road traffic sign in motion image based on genetic algorithm, in: Proceedings of the 1st International Conference on Machine Learning and Cybernetics, 2002, 83–86.
[22] R. González, R. Woods, Digital Image Processing, Addison-Wesley, 1993.
[23] P. Gil, S. Maldonado, H. Gómez, S. Lafuente, F. López, Traffic sign shape classification and localization based on the normalized FFT of the signature of blobs and 2D homographies, Signal Processing 88 (2008) 2943–2945.
[24] A. Soetedjo, K. Yamada, Traffic sign classification using ring partitioned method, IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences E88-A (9) (2005) 2419–2426.
[25] N. Otsu, A threshold selection method from gray-scale histogram, IEEE Transactions Systems, Man, and Cybernetics 8 (1978) 62–66.
[26] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (2) (1998) 121–167.
[27] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, New York, NY, USA, 2006.
[28] C. Chang, J. Chih, 2001. URL <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
[29] K.-M. Chung, W.-C. Kao, C.-L. Sun, L.-L. Wang, C.-J. Lin, Radius margin bounds for support vector machines with the RBF kernel, Neural Computation 15 (2003) 2643–2681.
[30] V. Vapnik, O. Chapelle, Bounds on error expectation for support vector machines, Neural Computation 12 (9) (2000) 2013–2036.
[31] T. Joachims, Estimating the generalization performance of a SVM efficiently, in: P. Langley (Ed.), Proceedings of ICML-00, Morgan Kaufmann , San Francisco, US, 2000, pp. 431–438.
[32] K. Duan, S. Sathiya, A. Poo, Evaluation of simple performance measures for tuning the SVM hyperparameters, Neurocomputing 51 (2003) 41–59.
[33] M. Clerc, J. Kennedy, The particle swarm-explosion, stability and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–73.
[34] J. Acevedo, S. Maldonado, P. Siegmann, P. Lafuente, S. Gil, Multi-class support vector machines based on arranged decision graphs and particle swarm optimization for model selection, Lecture Notes in Computer Science 4432 (2007) 238–245.
[35] I.C. Trelea, The particle swarm optimization algorithm: convergence analysis and parameter selection, Information Processing Letters 85 (6) (2003) 317–325.
[36] J. Platt, Probabilistic outputs for support vector machines and comparison to regularized likelihood methods, in: A. Smola, P. Bartlett, B. Schoelkopf, D. Schuurmans (Eds.), Advances in Kernel Methods – Support Vector Learning, 2000, pp. 61–74.
[37] J. Shawe-Taylor, N. Cristianini, Kernel Methods for Pattern Analysis, Cambridge University Press, New York, NY, USA, 2004.
[38] I. Guyon, J. Markhoul, R. Schwartz, V. Vapnik, What size test set gives good error rate estimates?, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1) (1998) 52–64
[39] GRAM, Video Sequence Including Ideas Exposed in CVIU Manuscript, URL<http://agamenon.tsc.uah.es/Investigacion/gram/CVIU>.

Please cite this article in press as: S. Maldonado Bascón et al., An optimization on pictogram identification for the road-sign recognition task using SVMs, Comput. Vis. Image Understand. (2009), doi:10.1016/j.cviu.2009.12.002