Optimizing color information processing inside an SVM network

J. Pasquet∗, G. Subsol∗∗, M. Derras∗ and M. Chaumont†∗

∗Berger-Levrault, Labège, France
†Univ. Nimes, France
‡LIRMM, Univ. Montpellier / CNRS, France

Abstract

Today, with the higher computing power of CPUs and GPUs, many different neural network architectures have been proposed for object detection in images. However, these networks are often not optimized to process color information. In this paper, we propose a new method based on an SVM network, that efficiently extracts this color information. We describe different network architectures and compare them with several color models (CIELAB, HSV, RGB...).

The results obtained on real data show that our network is more efficient and robust than a single SVM network, with an average precision gain ranging from 1.5% to 6% with respect to the complexity of the test image database. We have optimized the network architecture in order to gain information from color data, thus increasing the average precision by up to 10%.

Introduction

Object detection in aerial images is a complex task which is usually dealt with by a machine learning approach where classification is based on a single or cascade of SVM [1, 2]. However, it is hard to concatenate different features in a single vector without any normalization problem [3].

A way to overcome this difficulty is to distribute the different features in different inputs of a neural network and to use a classifier as output node. Many network architectures have recently been proposed to create deep convolutional neural networks [4, 5]. However, this kind of approach usually requires millions of parameters and thus a huge database to determine them. To solve this problem, we can replace each neuron of the network by a linear SVM which is known to converge faster [6]. Similar to a neuronal network, each SVM uses an activation function.

Currently most image databases are in color. But, surprisingly, most object detection algorithms use only a single color channel, which can be obtained by different methods [8], and then the resulting gray-level image is processing by the network. Color information from the image is then only very partially used.

In this paper, we first show experimentally that using all the color information may lead to a significant improvement in accuracy1. Then we discuss how to choose the network architecture in order to use color information in an optimal way. For this, we give an overview of our SVM network system and describe the experimental database in section 2. In section 3, we analyze the performance by using entire color information and studying the influence of the color space. In section 4, we describe several architectures for our network and then find and analyze the optimal one. Finally, we conclude and present future work.

SVM network and data

Our SVM network

Our object detection procedure classifies images based on the Histogram of Oriented Gradients descriptor [9]. First we normalize the bounding box of each object image of the training database to an image of constant size, as in [2]. Then we extract the HOG features in a sliding window on the normalized image of the object. Thus, for an object, we obtain a vector composed of many HOG features. To get a multi-resolution descriptor, we repeat the extraction step with different sizes of sliding windows, as in [10].

The classical method [2, 10] is to build a single large vector by concatenating vectors corresponding to the different sizes of sliding windows and then use a SVM classifier. But, in our case, we decided to use the different vectors as inputs of an SVM network [11]. To get better multi-resolution robustness, each input neuron learns on a single size. In addition, to introduce some invariance in our SVM network, we also create a set of input neurons which are randomly connected to part of the vector HOG in different resolutions [12].

Then a hidden layer is randomly connected to input neurons to get a better high-level abstraction. To break the linearity an asymmetric sigmoid function (see equation 1) is apply as an activation function after each SVM.

\[ f(x) = \frac{1}{1 + e^{-ax+\lambda}} \]  

(1)

Where \( \alpha \) and \( \lambda \) are two parameters computed with a validation database according to [7].

1We thank L. Deruelle and F. Bibonne (Berger Levrault, Labège, France) for providing access to image data and for fruitful discussions.
The output neuron is fully-connected to the last hidden layer. During the classification step, this neuron will return the probability for the input window to contain a target object.

**Our experimental database**

We are interested in the detection of urban objects in high-definition aerial images. More specifically, we focus on detecting man-hole covers [13] and tombs [14] for geo-localization purposes. In particular, tomb detection appears to be a very challenging problem as tombs vary substantially in appearance, color and size in aerial images. Moreover, vegetation, shadows created by the numerous buildings, people walking or utility vehicles may create many distortions and occlusions in the images. We thus selected this application for our tests.

For the training database, we used 19 industrial aerial images of cemeteries with a resolution of 2.5 cm/pixel, which contained about 4,500 tombs. For the validation database, we used 3 images containing about 750 tombs. To evaluate our algorithms, we used 2 cemetery images (about 700 tombs) and manually delineated the rectangular bounding boxes of tombs which are then considered as ground truth.

During the learning and test step, we use the High Performance Computing resources of HPC-LR\(^2\) in order to have a reasonable computational time.

During the evaluation step, detection results are given as a list of rectangular bounding boxes. To be considered as a correct detection, the area of overlap, noted \(A\) (see equation 2), between the detected boxes \(B_d\) and the ground truth \(B_g\) must exceed 58%.

\[
A = \frac{\text{Area}(B_d \cap B_g)}{\text{Area}(B_d \cup B_g)}
\]

Using all the color information

**How to integrate information in the network?**

Color information is defined in a color space \(S\) and is given as a set of \(S\) channels \(c\) (in general \(S=3\)). For example, \(c \in S = \{R, G, B\}\) for the RGB color space. In the following, \(f_c^r\) denotes the HOG vector computed for a given window for channel \(c\) at a resolution of \(r \in \{0,1, \ldots, N\}\), with \(N\) being the number of extracted windows of different sizes. Let \(p_c^r\) denote the output of a neuron which takes vector \(f_c^r\) as input.

The simplest idea consists of directly sending the \(f_c^r\) vectors inside the network. This means that each channel is processed separately and the number of input neurons is \(N\) times \(c\). The hidden layer allows the network to combine the different channels and manage the color information. For this purpose, we need to increase the number of hidden neurons and thus the overall complexity. Figure 1 shows this kind of architecture. We will call this architecture the *separate architecture* in the case of a color space with \(S = 2\) channels.

**Choice of color space**

We test the separate configuration network through five standard color spaces. The RGB space is the addition of three fundamental colors, i.e. red, green and blue to form all the colors. The luminance is a definition of the intensity level \((S = 1)\), \(Y = 0.21R + 0.71G + 0.07B\) according to [15]. \(CIE-LAB\) and \(CIE-LUV\) are two color spaces based on human perception. These models give optimal results in some image processing applications as in [16]. The \(HSV\) (Hue, Saturation and Value) color space is a cylindrical coordinate representation of the \(RGB\) model. It is very effective for image segmentation, as shown in [17].

*Optimal Parameters for SVM Network*

The best number of hidden SVN neurons and random neurons in the SVM network is an important parameter. If the number of random neuron is large enough the precision converges to a constant asymptote[12]. We thus took 400 random neurons in all the experiments. In Figure 2 we use a validation database which allows us to determine the number of hidden neurons. As expected, this number is directly linked to the size of first layer (input neurons) and thus to the color space used. The best number of hidden neurons is 300 and 600 for a color space of size \(S = 1\) and \(S = 3\) respectively.

**Experimental results**

First, we compare our network based on the *separate architecture* with the classical approach where we use an unique SVM which receives an input vector composed of concatenations of \(f_c^r\).

In Figure 3, the experiments show that for any color space, the SVM network outperforms a single SVM. We even obtain a
small average gain in precision of about 1.5% if we only use the
to the right

Figure 3. Comparison of the performance of our SVM network and a single

Figure 4. Comparison of the performance of our SVM network and a single

The Fusion architecture

The Maximum and Product architecture

A Network Architecture to Optimize Color Information Processing

As shown in the previous section, information from the RGB and
HSV model increases the performance. Nevertheless, compared
to the results obtained with one grey level (L or Y), results
with color are closed. The separate architecture is probably not
well enough designed to integrate the color information. This can
be explained because the separate architecture does not directly
create relationships between neurons with the same resolution. In
this section, to deal with this problem we introduce and test five
other architectures to process 3-channel color information.

The Fusion architecture

In this approach, we propose to concatenate the different
for a number of windows N and with the entire value of the
channel space S.

The advantage of this method is that each input neuron simulta-
eously treats the entire color space, so during the learning step input neurons will be able to adapt to the database. More-
over, the number of input neurons does not increase, so we do not
increase the size of the hidden layer and the computational time.
We call this the fusion architecture.

The Maximum and Product architecture

This method was inspired by the following paper [18]. In-
deed, using the orientation and magnitude of the gradient image,
we do not want to focus on the low variations. Moreover, we pro-
sue to only select signals with the highest variation. The first step
requires building the input neurons as in the separate architecture.
The second step is to connect all input neurons with the same res-

solution r to a specific max neuron. We define a max neuron as
a neuron which relays the highest signal without any activation
function (see equation 3).

The second layer contains all the max neurons and it also has
two advantages. Firstly, it selects only the highest channel inten-
sity independently of r and secondly, only the max neuron outputs
are used by the hidden layer. Schema 5 is an example of this ar-

max neuron, the product neurons perform a linear quantification as
defined in equation 4. This transformation is effective because the
product neurons returns different scores if few intensity channels
Figure 5. The maximum (and product by replacing $p_{\text{max}}$ by $p_{\text{prod}}$) architecture.

have stray variations. We call this the product architecture.

$$p_{\text{prod}}^r = \prod_{c \in \mathcal{S}} p_{c}^r$$

(4)

The PCA architecture

To choose the best color subset, the PCA is one classic alternative [8]. We assume that the best color subset is not always the same for each feature resolution. We thus have to fit a PCA projection noted $pca^r$, for each resolution $r$, Schema 6.

Like the maximum architecture, this architecture has two design steps. The first one consists in the building input neurons similarly to the separate architecture. The second step consists to connect all input neurons with the same resolution $r$ to a specific PCA neuron. We then calculate the eigenvalues after the learning of the input neurons. Each PCA neuron returns only the first principal component which is a simple scalar product. To break the linearity, we add an asymmetric sigmoid activation function as for SVM neuron. This architecture is called PCA architecture.

The Stack architecture

In the fusion architecture, each input neuron treats all vectors from $\mathcal{S}$. More generally, we may want to use a set of feature vectors which do not belong to the same space as for example SURF and HOG vectors. This requires first to scale each feature in order than the network can process all of them in a consistent way.

For this, the fusion architecture is refined by integrating a sub-layer which takes the features $f_c^r$ as separate inputs and computes $p_c^r$. These values allow to compute a new feature vector $f_{\text{weight}}^r$ which is a weighted combination of $f_c^r$ for a given resolution $r$ by:

$$f_{\text{weight}}^r = p_{c_1}^r f_{c_1}^r, p_{c_2}^r f_{c_2}^r, \ldots, p_{c_S}^r f_{c_S}^r$$

(5)

The low level input vectors will be used for the higher neurons [19] with a rescaling in this new architecture called stack architecture, as in Figure 7.

**Experimental Results**

In Figure 8, the maximum architecture does not enhance the results. Indeed, in natural images the intensity in the three channel RGB often fluctuates at the same time. The addition of color information is then weak compared to luminance. However, the product architecture is more sensitive to any fluctuations in different color components and thus has about 7% better precision for a recall range of 55% to 82%. In the same idea, the precision of the PCA architecture is similar or less than the precision of the product architecture.

The fusion architecture also increases the performance for a recall range of 50% to 82%. But this architecture can only be effective for features of the same type, unlike other proposed architectures.

The stack architecture increases the precision by up to 10% compared to the separate architecture. In fact, this architecture combines all advantages: it has a minimal number of hidden neurons like the maximum architecture and all of the information from channels is kept like the separate architecture. Moreover, there is no requirement to give an explicit formula to transform the different channels to only one as in the maximum or product architecture. In addition, the stack architecture can be used to combine descriptors from different feature spaces.

However, the disadvantage of this method lies in the computational cost during the learning step and during the testing step. To deal with the calculation time during the evaluation step the activation path could be used which reduced the requiring time by 80% [12].

**Conclusion and Future Work**

In this paper, we first compare a single SVM-based approach to an SVM network approach. We show that an SVM network
outperforms the single SVM by an average precision gain ranging from 1.5% to 6%. In addition, we notice that using the three color components does not significantly increase the single SVM results. On the contrary, we can see that an SVM network is able to extract more information from the different color components and thus increases the final performance.

We then introduce different network architectures in order to effectively combine the features of each color component. The best proposed takes into account all the different features and normalizes them using a smart weight. We show that the so-called stack architecture outperforms other architectures by an average gain in precision ranging from 4% to 10%.

Future works will deal with the addition of features such as color components, SURF, SIFT or pixel intensity where the stack architecture could be used. But more tests are required to validate the real effectiveness of this design.

Another extension of our work would be to extend the stack architecture to a deep neural network. Thus the weighted combination of $f^2_i$ will be learned by back propagation.

### Acknowledgement

The authors would like to thank the Berger-Levrault group for supporting this research. Berger-Levrault is a French public regulation expert that addresses healthcare and local public administrations. With almost 60 thousands customers and more than 1,000 employees, the Berger-Levrault group is a key enabler for the development of e.Government in France. Bringing people/citizens/patients and their public administrations closer is a key focus of the Berger-Levrault group.

### References


[7] Jing Gao and Pang ning Tan, “Converting output scores from outlier detection algorithms into probability estimates,” in Proc. of the Sixth IEEE Int. Conf. on Data Mining (ICDM06), 2006, pp. 212–221.


detection of color images using CIELAB color space and edge detec-
tors,” in *Emerging Trends in Robotics and Communication Technolo-
393–397.


[18] D. Aldavet, A. Ramisa, R. López de Mántaras, and R. Toledo,
“Real-Time Object Segmentation using a Bag of Features Approach,”
in *13th International Conference of the ACIA, L’Espluga de Francolí,
Catalonia, Spain, 2010*, IOS Press, IOS Press.

[19] Kai Ming Ting and Ian H. Witten, “Stacked generalization: when
does it work,” in *Procs. International Joint Conference on Artificial
Intelligence*. 1997, pp. 866–871, Morgan Kaufmann.

Author Biography

Jérôme Pasquet received a master degree in computer science from
the university of Montpellier II, France, in 2013. He is currently work-
ing toward the PhD degree in LIRMM (Montpellier Laboratory of Infor-
matics, Robotics and Microelectronics). His research interests are urban
objects detection and segmentation in aerial photography.

Gérard Subsol obtained a Ph.D. Thesis in Computer Science in
1995. He was successively Ph.D. student, then Expert-Engineer at IN-
RIA Sophia Antipolis, Research Engineer at University of Avignon and
R&D Engineer with the start-up company Intrasense. Since 2006, he has
been a CNRS Researcher at LIRMM located in the South of France. He is
currently working on several applications of 2D and 3D image processing.

Mustapha Derras obtained a Ph.D. Thesis in Computer Science in
1993. He is currently Director of Technology, Innovation and Research
of Berger-Levrault. Previously he held positions with the aim of creat-
ing innovative new products as Business Unit Director Catia, Dassault
Systèmes, Marketing and R&D at TIMEG, Director of Business Unit CAT
at Cadence Design Systems, software development architect at General
Electric Medical Systems and project Manager at CLAAS where he com-
pleted his mobile robotics and image processing studies.

Marc CHAUMONT received his Engineer Diploma in Computer
Sciences at the INSA (National Institute of Applied Sciences) of Rennes,
France in 1999, his Ph.D. at the IRISA Rennes in 2003, and his HDR
(“Habilitation à Diriger des Recherches”) at the University of Montpel-
lier in 2013. Since September 2005, he is an Assistant Professor in the
LIRMM laboratory of Montpellier and the University of Nimes. His re-
search areas are multimedia security (steganography, watermarking, dig-
ital forensics, video & image compression) and segmentation & tracking
in images and videos. He is member of the TC of IEEE SPS - Informa-
tion Forensics and Security for the period 2015-2017. He was program
chair of ACM IH&MMSec’2013. He is reviewer for more than 20 jour-
nals (IEEE TIFS, IS&T JEI, ...) and for more than 10 conferences (EI
MWSF, IEEE WIFS, ACM IH&MMSec, IEEE ICIP, ...).