Generic Visual Categorization Using Weak Geometry

Conference Paper · January 2006

DOI: 10.1007/11957959_11 · Source: DBLP

4 authors, including:

Gabriela Csurka
Naver Labs Europe
188 PUBLICATIONS **10,431** CITATIONS
SEE PROFILE

Jutta Willamowski
Naver Labs
74 PUBLICATIONS **4,448** CITATIONS
SEE PROFILE

Generic Visual Categorization Using Weak Geometry*

Gabriela Csurka, Christopher R. Dance, Florent Perronnin, and Jutta Willamowski

Xerox Research Centre Europe, 6 chemin de Maupertuis
38240, Meylan France
Firstname.Lastname@xrce.xerox.com

February 19, 2014

Abstract

In the first part of this chapter we make a general presentation of the bag-of-keypatches approach to generic visual categorization (GVC). Our approach is inspired by the bag-of-words approach to text categorization. This method is able to identify the object content of natural images while generalizing across variations inherent to the object class. To obtain a visual vocabulary insensitive to viewpoint and illumination, rotation or affine invariant orientation histogram descriptors of image patches are vector quantized. Each image is then represented by one visual word occurrence histogram. To classify the images we use one-against-all SVM classifiers and choose the best ranked category. The main advantages of the method are that it is simple, computationally efficient and intrinsically invariant. We obtained excellent results as well for multi-class categorization as for object detection.

In the second part we improve the categorizer by incorporating geometric information. Based on scale, orientation or closeness of the keypatches we can consider a large number of simple geometrical relationships, each of which can be considered as a simplistic classifier. We select from this multitude of classifiers (several millions in our case) and combine them effectively with the original classifier. Results are shown on a new challenging 10 class dataset.

1 Introduction

The proliferation of digital imaging sensors in mobile phones and consumer-level cameras is producing a growing number of large digital image collections and increasing the pervasiveness of images on the web and in other documents. To search and manage such collections it is useful to have access to high-level information about objects contained in the images. We are therefore interested in recognizing several objects or image categories within a multi-class categorization system, but not in the localization of the objects which is unnecessary for most applications involving tagging and search. In this chapter we describe a generic visual categorization (GVC) system which is sufficiently generic to cope with many object types simultaneously and which can readily be extended to new categories. It can handle variations in view, background clutter, lighting and occlusion as well as intra-class variations.

Before describing the approach we underline the distinction of visual categorization from three related problems:

- *Recognition*: This concerns the identification of particular object instances. For instance, recognition would distinguish between images of two structurally distinct cups, while categorization would place them in the same class.

*Published in Lecture Notes in Computer Science Volume 4170, 2006, pp 207-224, Springer Berlin Heidelberg, DOI 10.1007/11957959_11

- *Content Based Image Retrieval*: This refers to the process of retrieving images on the basis of low-level image features, given a query image or manually constructed description of these low-level features. Such descriptions frequently have little relation to the semantic content of the image.

- *Detection*: This refers to deciding whether or not a member of one visual category is present in a given image. While it would be possible to perform generic categorization by applying a detector for each class of interest to a given image, this approach becomes inefficient given a large number of classes. In contrast to the technique proposed in this paper, most existing detection techniques require precise manual alignment of the training images and the segregation of these images into different views, neither of which is necessary in our case.

Our generic visual categorization system is a bag-of-keypatches approach which was motivated by an analogy to learning methods using the bag-of-words representation for text categorization [9, 23, 13]. In the bag-of-words representation, a text document is encoded as a histogram of the number of occurrences of each word. Similarly, one can characterize an image by a histogram of visual word counts. The visual vocabulary provides a "mid-level" representation which helps to bridge the semantic gap between the low-level features extracted from an image and the high-level concepts to be categorized [1]. However, the main difference from text categorization is that there is no given vocabulary for images. Instead we generate a visual vocabulary automatically from a training set.

The idea of adapting text categorization approaches to visual categorization is not new. Zhu *et al* [26] investigated the vector quantization of small square image windows, which they called keyblocks. They showed that these features produced more "semantics-oriented" results than color and texture based approaches, when combined with analogues of the well-known vector-, histogram-, and n-gram-models of text retrieval. In contrast to our approach, their keyblocks do not possess any invariance properties. Our visual vocabulary [4] is obtained by clustering rotation or affine invariant orientation histogram descriptors using the K-means algorithm. In a similar way Sivic and Zisserman [22] used vector quantized SIFT descriptors of shape adapted regions and maximally stable regions to localize all the occurrences of a given object in a video sequence.

In these cases each centroid corresponds to a visual word and, to build a histogram, each feature vector is assigned to its closest centroid. In [8], Hsu and Chang argue that the clusters obtained with K-means have a high correlation with the low-level features but a weak correlation with the concepts. They devised a visual cue cluster construction based on the information bottleneck principle. More recently soft clustering using Gaussian Mixture Model (GMM) was proposed as an alternative to K-means [5, 19]. In this case, a low-level feature is not assigned to one visual word but to all words probabilistically, resulting in a continuous histogram representation.

Others have explored the post-processing of K-means clustering. For instance Sivic *et al* [21] use Probabilistic Latent Semantic Analysis (PLSA) to discover topics in a corpus of unlabelled images. Test images were then categorized based on the most relevant topic.

In order to improve the accuracy of our system we further exploit a boosting approach based on keypatches and simple geometrical relationships (similar scales, similar orientation, closeness) between them. We chose to adopt the boosting approach because there are many possible geometric relationships and boosting offers an effective way to select from this multitude of possible features. Boosting was used with success in [16] to detect the presence of bikes, persons, cars or airplanes against background. However their approach differs from ours as they do not include any geometry and consider every appearance descriptor without considering a vocabulary.

The main advantage of our approach is that geometric constraints are introduced as weak conditions in contrast to others such as [6, 11], where due to the use of relatively strong geometric models, such previous methods requires the alignment and segregation of different views of objects in the dataset.

Several other categorization approaches have recently been developed that are based on image segmentation [2, 12, 17, 3], rather than the interest point descriptors. In [2] geometry has been included through generative MRF models of neighboring relations between segmented regions. In contrast we prefer to take a discriminative classifier approach in order to optimize overall accuracy.

The remainder of this paper is organized as follows: section 2 describes the original bag of keypatches approach; in section 3 we introduce an alternative based on the boosting framework; in section 4 we

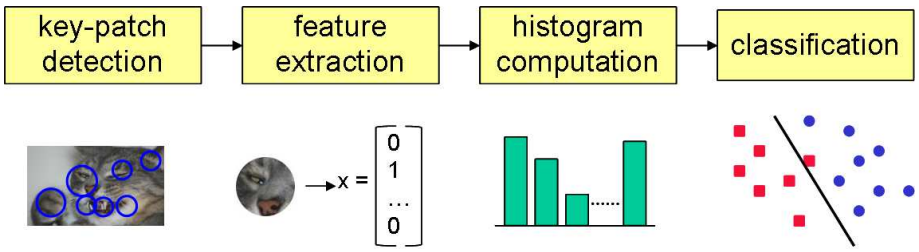

Figure 1: The main steps of the bag-of-keypatches approach.

then describe how to incorporate weak geometry in the boosting approach; we present experimental results in section 5 and conclude in section 6.

2 The Bag-of-Keypatch Approach

The main steps of the bag-of-keypatches approach introduced in [4] are as follows (see also Figure 1):

- Detect image patches and assign each of them to one of a set of predetermined clusters (a visual vocabulary) on the basis of their appearance descriptors.
- Construct a bag-of-keypatches by counting the number of patches assigned to each cluster.
- Apply a multi-class classifier, treating the bag-of-keypatches as the feature vector, and thus determine which categories to assign to the image. The multi-class classifier is built from a combination of one-against-all classifiers.

The extracted descriptors of image patches should be invariant to the variations that are irrelevant to the categorization task (viewpoint change, lighting variations and occlusions) but rich enough to carry all necessary information to be discriminative at the category level. We used Lowe's SIFT approach [14] to detect and describe image patches. This produces scale-invariant circular patches that are associated with 128-dimensional feature vectors of Gaussian derivatives. While in [4] we used affine invariant elliptical patches [15], similar performance was obtained with circular patches. Moreover, the use of circular patches makes it simpler to deal with geometric issues.

The visual vocabulary was constructed using the K-means algorithm applied to a set of over 10000 patches obtained from a set of images that was completely independent from the images used to train or test the classifier. We are not interested in a *correct clustering* in the sense of feature distributions, but rather in an accurate categorization. Therefore, to overcome the initialization dependence of K-means, we run it several times with different initial cluster centers and select the final clustering giving the highest categorization accuracy using an SVM classifier (without any geometric properties) on a subset of the dataset.

For categorization we use the SVM which finds the hyperplane that separates two-class data with maximal margin [25]. The margin is defined as the distance of the closest training point to the separating hyperplane. The SVM decision function can be expressed as:

$$f(\mathbf{x}) = \text{sign}(\sum_i y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b)$$

where $\mathbf{x}_i$ are the training features from data space and $y_i \in \{-1, 1\}$ is the label of $\mathbf{x}_i$. The parameters α_i are zero for most i, so the sum is taken only over a selected set of $\mathbf{x}_i$ known as support vectors. It can be shown that the support vectors are those feature vectors lying nearest to the separating hyperplane.

In this chapter, the input features $\mathbf{x}_i$ are the binned histograms formed by the number of occurrences of keypatches in the input image. K is a kernel function corresponding to an inner product between two transformed feature vectors, usually in a high and possibly infinite dimensional space. In the experiments described here we used a linear kernel, which is the dot product of $\mathbf{x}$ and $\mathbf{x}_i$.

In order to apply the SVM to multi-class problems we took the one-against-all approach. Given an m-class problem, we trained m SVM's, each of which distinguishes images from some category i from images from all the other $m-1$ categories j not equal to i. Given a query image, we assigned it to the class with the largest SVM output.

3 The Boosting Approach

An alternative to the SVM classifier is the boosting approach. Here we exploit the generalized version of the AdaBoost algorithm described in [20]. Boosting is a method of finding an accurate classifier H by combining M simpler classifiers h_m:

$$H(\mathbf{x}) = \Big(\sum_{m=1}^{M} \alpha_m h_m(\mathbf{x}) \Big) / \Big(\sum_{m=1}^{M} \alpha_m \Big). \tag{1}$$

Each simpler classifier $h_m(\mathbf{x}) \in [-1, 1]$ needs only to be moderately accurate and is therefore known as a *weak classifier*. They are chosen from a classifier space to maximize correlation[1] of the predictions and labels:

$$r_m = \sum_i D^m(i) h_m(\mathbf{x}_i) y_i,$$

where $D^m(i)$ is a set of weights (distribution) over the training set. At each step the weights are updated by increasing the weights of the incorrectly predicted training examples:

$$D^{m+1}(i) = D^m(i) \exp\{-\alpha_m y_i h_m(\mathbf{x}_i)\} / Z_m \tag{2}$$

where

$$\alpha_m = \frac{1}{2} \log \frac{1 + r_m}{1 - r_m} \tag{3}$$

and Z_m is a normalization constant, such that $\sum_i D^{m+1}(i) = 1$.

To define the weak classifiers we consider the same inputs as for the SVM, i.e. the binned histograms $\mathbf{x}_i$. The simplest keypatch-based weak classifier $h^{k,T}$ counts the number of patches whose SIFT features belong to cluster k, which is equivalent to comparing $\mathbf{x}_i^k$ to some threshold T. If this number is at least T, then the classifier output is 1, otherwise -1:

$$h^{k,T}(\mathbf{x}_i) = \begin{cases} 1 & \text{if} \quad \mathbf{x}_i^k \geq T \\ -1 & \text{otherwise} \end{cases} .$$

We may build similar weak classifiers $h^{kl,T}$ from a pair of keypatch types k, l. If at least T keypatches of both types are observed, then the classifier output is 1:

$$h^{kl,T}(\mathbf{x}_i) = \begin{cases} 1 & \text{if} \quad \mathbf{x}_i^k \geq T \text{ and } \mathbf{x}_i^l \geq T \\ -1 & \text{otherwise} \end{cases} .$$

In practice we select weak classifiers by searching over a predefined set of thresholds such as $\{1, 5, 10\}$. The opposite weak classifier $h^{k,\bar{T}}$ can also be defined by inverting the inequality ($\mathbf{x}^k < T$). Four such definitions are possible for pairs of keypatches $h^{kl,T}$, $h^{kl,\bar{T}}$, $h^{kl,T\bar{T}}$ and $h^{kl,\bar{T}T}$, e.g:

$$h^{kl,T\bar{T}}(\mathbf{x}_i) = \begin{cases} 1 & \text{if} \quad \mathbf{x}_i^k \geq T \text{ and } \mathbf{x}_i^l < T \\ -1 & \text{otherwise} \end{cases} .$$

In practice, we search over the full set of different possibilities when working with weak classifiers and refer to them collectively as h^k and h^{kl}. Obviously, it would be possible to further extend the definition for pairs to applying a different threshold to each keypatch type (T_k and T_l). In practice, we avoid this as it results in a prohibitively large number of possible weak classifiers.

[1] This is equivalent to minimizing the weighted training error which is equal to $(1 - r_m)/2$.

4 Incorporating Geometric Information

In this section we describe some ways to construct geometric weak classifiers. As input, we assume each patch i in a query image has been labeled according to its appearance via the index of the cluster centre k_i to which it is assigned. Each patch is associated with its orientation θ_i and a ball (circular patch) B_i which has center position p_i and scale σ_i.

A simple way to incorporate geometrical information in weak classifiers depending on one keypatch is to threshold the number of interest points belonging to a cluster k and having a particular *orientation*:

$$h_\theta^{k,T}(I) = \begin{cases} 1 & \text{if} \quad \exists\theta \text{ such that } \left|\{i \in \mathcal{P}_I : k_i = k, \theta_i = \theta\}\right| \geq T \\ -1 & \text{otherwise} \end{cases}$$

where $\left|A\right|$ denotes the cardinality of the set A and $\mathcal{P}_I$ denotes the set of patches in image I.

Note that a large number of different orientations are produced by the interest point detectors. Therefore we exploit a coarse quantization of the orientations into eight bins. Two keypatches are considered to have the same orientation if they fall into the same bin[2]. This does not constitute exact orientation invariance, as a small rotation could cause two keypatches in one bin to move to different bins. However, this approach is more efficient than directly measuring and thresholding the difference in orientations $\|\theta_i - \theta_j\|$ between pairs of keypatches.

Likewise, we define sets of weak classifiers that count the number of keypatches with the same *scale* or a set that count patches with both the same *scale and orientation*. The scale bins are selected with logarithmic spacing, in order to approximate scale invariance. Collectively[3] these classifiers are denoted by $h_\theta^k, h_\sigma^k, h_{\sigma,\theta}^k$.

Another way to incorporate geometry is to count the *number of interest points in the ball* around a keypatch of a given type. This count is made irrespective of the type of keypatches in the ball. As with the other weak classifiers, this property is invariant to shift, scaling and rotation. In a given image, there may be multiple keypatches of a given type containing different numbers of points. We define h_B^k in terms of the keypatch of type k with the maximum number of points in its ball:

$$h_B^{k,T}(I) = \begin{cases} 1 & \text{if} \quad \exists i \text{ such that } k_i = k \text{ and } \left|\{j \in \mathcal{P}_I : p_j \in B_i\}\right| \geq T \\ -1 & \text{otherwise} \end{cases}$$

where $p_j \in B_i$ means that the center of the patch j is inside of the ball B_i defined by the patch i.

Taking two types of keypatches k and l into consideration, there are more ways to introduce geometry. Classifiers based on common scale or orientation can be extended in two obvious ways. Firstly we can require that the patches of type k and those of type l have *identical* scale and/or orientation, giving $h_{\sigma=}^{kl}, h_{\theta=}^{kl}, h_{\sigma\theta=}^{kl}$. Alternatively we can allow each type to have their own independent scales or orientations, giving $h_\sigma^{kl}, h_\theta^{kl}, h_{\sigma\theta}^{kl}$. The latter corresponds to a Boolean combination of single point classifiers, e.g. h_σ^k and h_σ^l.

A weak classifier h_B^{kl} can be constructed similarly to h_B^k that checks for the existence of a pair of interest points labeled k, l such that both of them have at least T interest points inside their balls.

We additionally consider five other ways of exploiting the position information associated with patches:

- $h_{k\in l}$ tests if there are at least T keypatches labeled l which contain an interest point labeled k within their ball.
- $h_{k\subset l}$ tests if there are at least T keypatches labeled l whose balls contain the whole ball of an interest point labeled k.
- $h_{k\cap l}$ tests if there are at least T keypatches labeled l whose balls intersect with the ball of at least one interest point labeled k.
- $h_{k\propto l}$ tests if there are at least T keypatches labeled l such that their closest neighboring interest points in the image are labeled k.

[2]The equality in the notation $\theta_i = \theta$ should be interpreted in this way.

[3]Considering similar threshold reversals as for h^k and h^{kl}, e.g. $h_\theta^{k,\bar{T}}$ and $h_{\sigma,\theta}^{kl,\bar{T}T}$.

$h^{y,5}_{\sigma}$, $h^{y,4}_{\sigma\theta}$, $h^{ry,2}_{\sigma}$, $h^{ry,5}_{\theta}$, $h^{ry,2}_{\sigma\theta}$, $h^{1}_{y\cap r}$ and $h^{1}_{y\in r} = 1$
$h^{r,6}$, $h^{y,6}_{\theta}$, $h^{ry,6}$, $h^{ry,1}_{\sigma^{=}}$, $h^{ry,1}_{\theta=}$, $h^{1}_{\sigma\theta=}$ and $h^{1}_{r\subset r} = -1$

Figure 2: Examples of weak classifiers on a typical image for keypatches of type r, y (red or yellow). For clarity, only the patches of type r and y are shown. In these examples, the threshold T on which the weak classifiers depend has been chosen as large as possible for output 1 (first row) and as small as possible for output -1 (second row).

- $h_{k\in\aleph^N_l}$ tests if there are at least T keypatch labeled l such that there exist a keypatch labeled k among its N closest neighbors.

The set of weak classifiers we considered is summarized in Table 1 and Figure 2 illustrates some of them. Of course there are a lot of other possibilities that could be experimented with.

5 Results

This section presents some results from our experiments. First we compare our bag-of-keypatch approach with the method described in [6]. Therefore we used the object classes from their FPZ dataset that are freely available, i.e. five object classes - 1074 airplane side images, 651 car rear images, 720 car side images, 450 frontal face images, and 826 motorbike side images - and a set of 451 background images.

The second set of experiments were done on a more challenging in-house dataset. This test was made to test larger number of classes, more variable poses and intra-class variations and significant amounts of background clutter. The images have resolutions between 0.3 and 2 mega-pixels and were acquired with a diverse set of cameras. The images are color but only the luminance component is used in our method. They were gathered by XRCE and Graz University. This dataset[4] contains 3084 images from 10 categories. The number of images per class are: bikes (237), boats (434), books (270), cars (307), chairs (346), flowers (242), phones (250), road signs (211), shoes (525) and soft toys (262). Figure 3 shows some images from this database.

We used the confusion matrix (4) to evaluate the multi-class classifiers and the overall correct rate

[4]The dataset is publicly available on ftp://ftp.xrce.xerox.com/pub/ftp-ipc

Table 1: Complete list of weak classifiers investigated. $p \propto q$ indicates that p is the closest point to q and $\aleph_{p_j}^N$ is the set of the N closest neighbors of p_j.

h	$h = \begin{cases} 1 & \text{if this quantity} \geq T \\ -1 & \text{otherwise} \end{cases}$	h	$h = \begin{cases} 1 & \text{if this quantity} \geq T \\ -1 & \text{otherwise} \end{cases}$
$h_{\sigma}^{k,T}$	$\max_{\sigma} \big\lvert\{i : k_i = k, \sigma_i = \sigma\}\big\rvert$	$h_{\sigma\theta}^{k,T}$	$\max_{\sigma,\theta} \big\lvert\{i : k_i = k, \sigma_i = \sigma, \theta_i = \theta\}\big\rvert$
$h_{\sigma}^{kl,T}$	$\min_{u\in\{k,l\}} \max_{\sigma} \big\lvert\{i : k_i = u, \sigma_i = \sigma\}\big\rvert$	$h_{\sigma\theta}^{kl,T}$	$\min_{u\in\{k,l\}} \max_{\sigma,\theta} \big\lvert\{i : k_i = u, \sigma_i = \sigma, \theta_i = \theta\}\big\rvert$
$h_{\theta}^{k,T}$	$\max_{\theta} \big\lvert\{i : k_i = k, \theta_i = \theta\}\big\rvert$	$h_{\sigma\theta=}^{kl,T}$	$\max_{\sigma,\theta} \min_{u\in\{k,l\}} \big\lvert\{i : k_i = u, \sigma_i = \sigma, \theta_i = \theta\}\big\rvert$
$h_{\theta}^{kl,T}$	$\min_{u\in\{k,l\}} \max_{\theta} \big\lvert\{i : k_i = u, \theta_i = \theta\}\big\rvert$	$h_{k\in l}^{T}$	$\big\lvert\{j : k_j = l, \exists k_i = k, p_i \in B_j\}\big\rvert$
$h_{\sigma=}^{kl,T}$	$\max_{\sigma} \min_{u\in\{k,l\}} \big\lvert\{i : k_i = u, \sigma_i = \sigma\}\big\rvert$	$h_{k\subset l}^{T}$	$\big\lvert\{j : k_j = l, \exists k_i = k, B_i \subset B_j\}\big\rvert$
$h_{\theta=}^{kl,T}$	$\max_{\theta} \min_{u\in\{k,l\}} \big\lvert\{i : k_i = u, \theta_i = \theta\}\big\rvert$	$h_{k\cap l}^{T}$	$\big\lvert\{j : k_i = l, \exists k_i = k, B_i \cap B_j \neq \emptyset\}\big\rvert$
$h_{B}^{k,T}$	$\max_{i} \big\lvert\{j : k_i = k, p_j \in B_i\}\big\rvert$	$h_{k\propto l}^{T}$	$\big\lvert\{j : k_j = l, \exists k_i = k, p_i \propto p_j\}\big\rvert$
$h_{B}^{kl,T}$	$\max_{i} \min_{u\in\{k,l\}} \big\lvert\{j : k_i = u, p_j \in B_i\}\big\rvert$	$h_{k\in\aleph_l^N}^{T}$	$\big\lvert\{j : k_j = l, \exists k_i = k, p_i \in \aleph_{p_j}^N\}\big\rvert$

(5) for the object detection:

$$M_{ij} = \frac{\big\lvert\{I \in \mathbf{C}_j : H_i(I) \geq H_m(I), \forall m\}\big\rvert}{\big\lvert\mathbf{C}_j\big\rvert}, \tag{4}$$

and

$$R = 1 - \frac{\sum_{j=1}^{N_c} \big\lvert\mathbf{C}_j\big\rvert M_{jj}}{\sum_{j=1}^{N_c} \big\lvert\mathbf{C}_j\big\rvert} \tag{5}$$

where N_c is the number of considered classes, $i, j \in \{1, \cdots, N_c\}$, $\mathbf{C}_j$ is the set of test images from category j and $H_m(I)$ is the real output of the classifier H_m which was trained to distinguish class m from the rest of the classes.

5.0.1 Vocabulary size.

There exist methods allowing to automatically select the number of clusters for K-means. For example, Pelleg *et al* [18] use cluster splitting, where the splitting decision depend on the Bayesian Information Criterion. However, in the present case we do not really know anything about the density or the compactness of our clusters. Moreover, we are not even interested in a "correct clustering" in the sense of feature distributions, but rather in accurate categorization. We therefore simply compare error rates for different values of K.

Figure 4 presents the overall error rates using the bag-of-keypatches approach on our in-house dataset as a function of the number of clusters K. Each point in Figure 4 is the "best"[5] of 10 random trials of

[5]Best in the sense of lowest empirical risk in categorization [25].

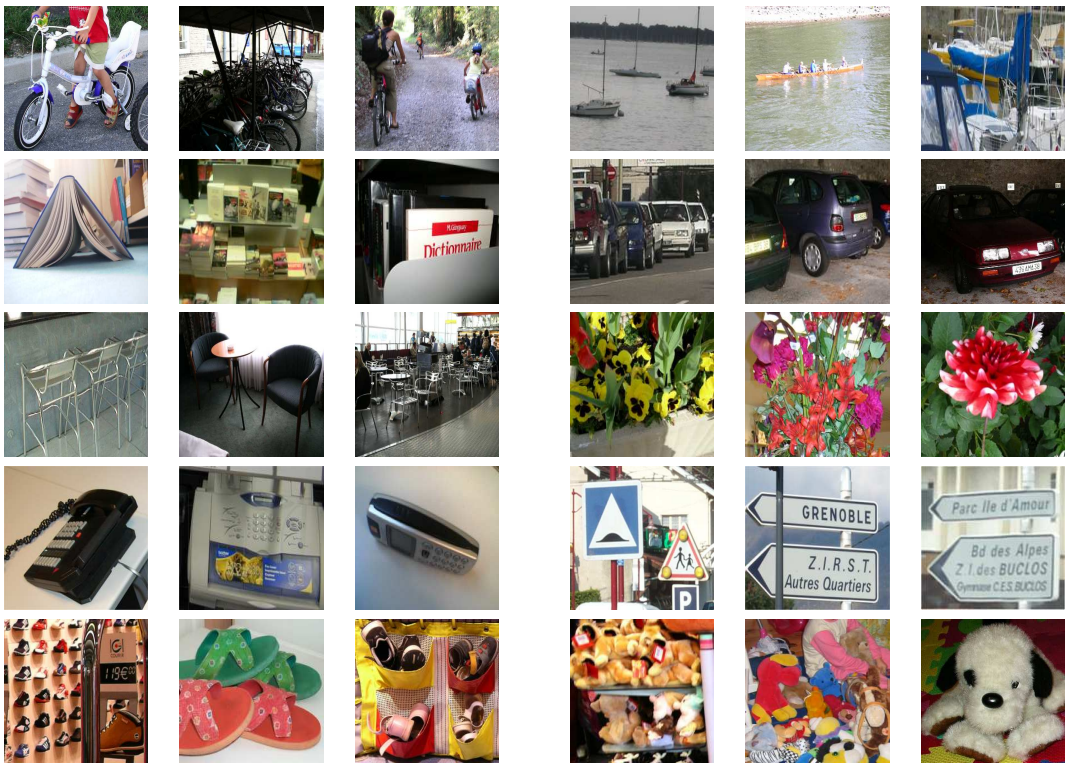

Figure 3: Examples from our 10 class dataset.

Table 2: Correct rates for all classes obtained in 2-fold cross-validation at the equal error rate point by Fergus *et al* (FPZ) with our bag-of-keypatches method (SVM_i) and a PLSA based approach ($PLSA$) described in [21]. The best results for each class are shown in bold face.

method	Airplanes	Cars(rear)	Cars(side)	Faces	Motorbikes
FPZ	90.2	N/A	**88.5**	96.4	92.5
SVM_1	**97.1**	**98.6**	87.3	**99.3**	**98**
SVM_2	96.4	97.9	86.1	98.9	97.3
$PLSA$	96.6	88.1	N/A	94.7	84.6

K-means. We can notice that the error rate only improves slightly as we move from k = 1000 to k = 2500. We therefore assert that k = 1000 presents a good trade-off between accuracy and speed and in all of our experiments we worked with the "best" vocabulary of size 1000.

5.0.2 Object Detection.

Table 2 compares our results with the ones obtained by Fergus *et al* as far as they are available from their paper [6] on the FPZ dataset. They were obtained using 2-fold cross-validation and the correct rates reported correspond to the equal error operating point. As they did, we train our classifiers to recognize foreground images, i.e. images belonging to the considered class, and reject background images. The difference between SVM_1 and SVM_2 is that to build the visual vocabulary (K-means) in the former case we used a subset of images from the FPZ database and in the latter case a completely independent image set. We can observe only a slight difference between the performances of SVM_1 and SVM_2, showing a low influence of the initial sample feature set on the classification results.

Except for cars (side) all the classifiers trained with our method perform much better, no matter

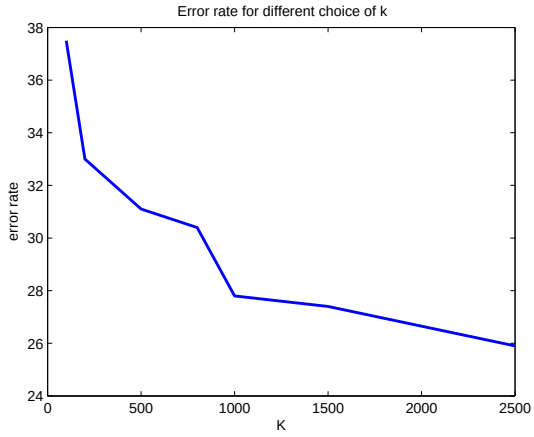

Figure 4: The lowest overall error rate (percentage) found for different choices of k.

which set of keypatches we use. The small difference on the cars (side) dataset is probably not significant. One possible reason why we do not perform so well on this category is that the cars (side) images are small and contain few keypatches (only about 50 keypatches compared with 500-1000 for the other classes).

Figure 5.0.2 shows the ROC curves for the classifiers obtained for SVM_1 with the different classes using 2-fold cross-validation. It shows that even for classifiers with a very small false positive rate the recall is very high.

5.0.3 Multi-class Classifier.

Tables 3 and 4 report the results we obtain using our method for training a multi-class classifier on the above mentioned five-class dataset. Table 3 shows the results with 2-fold cross-validation. This allows to compare the results with those from the object detection case (Table 2). It shows that in all cases except cars (side) the correct rates observed in the multi-class case are inferior to those obtained in the object detection case. Again this might be linked to the small number of keypatches present in the images belonging to this category.

Table 3: Overall correct rates for all classes obtained with 2-fold cross-validation with the different keypatch sets.

method	Airplanes	Cars(rear)	Cars(side)	Faces	Motorbikes
SVM_1	96.4	97.1	97.1	92.4	92.4
SVM_2	94.4	94.6	97.3	89.8	90.5
$SREZF1$	95.2	98.1	N/A	94	83.6
$SREZF2$	97.5	99.3	N/A	99.5	96.5

In two last rows of Table 3 ($SREZF1$ and $SREZF2$) we show the recent results obtained by Sivic et al [21] on this dataset using PLSA. They do not used cars (side) images. The inclusion of cars (side) is belived to confuse the classifier and significantly increase error rates.

They first used the training images (one fold) plus about 200 background images without their label and searched for 7 topics. In this way the PLSA discovered 3 topics related to background content and 4 topics corresponding to the 4 categories. In $SREZF1$ the test images were assigned to the most probable of the 7 topics. In $SREZF2$ test images were assigned only to the most probable of the 4 topics corresponding to categories (excluding the background topic from the ranking).

The results shows that using PLSA for automatic topic detection is promising and has the advantage that it do not need individual labeling of images. However it is difficult to judge how it scales with

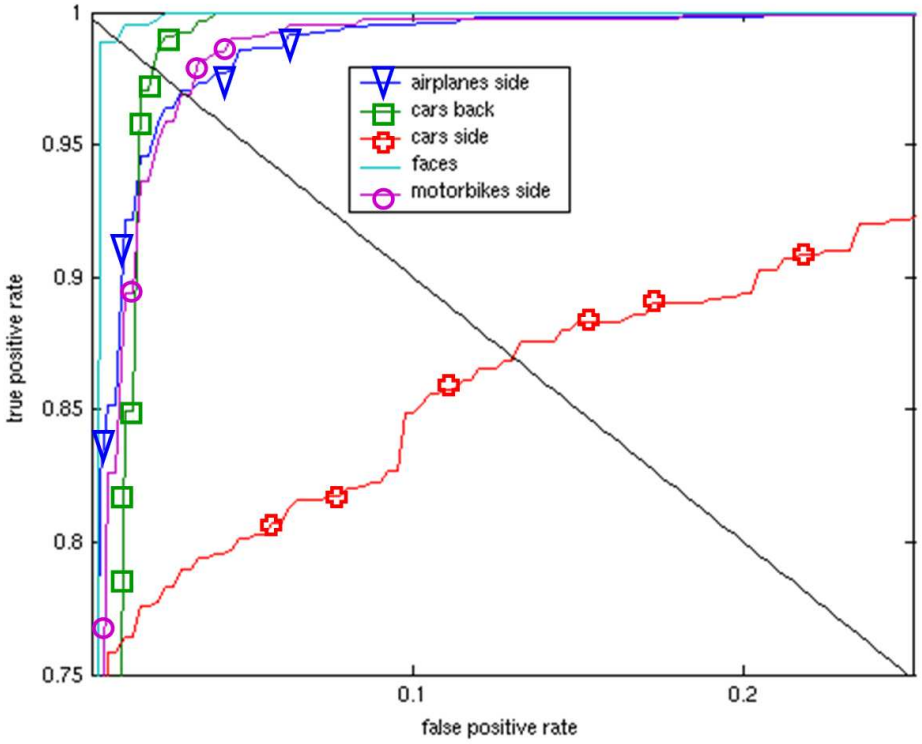

Figure 5: Zoomed-in view of the ROC curves obtained with keypatch set 2. The x-axis corresponds to false positive rate (1-precision), while the y-axis corresponds to the true positive rate (recall). The diagonal indicates the equal error rate (false negative rate = false positive rate).

the number of categories and as shown in Table 2 it works less well for binary classification against background.

Table 4: Confusion matrix for SVM_1 with 10-fold cross validation using a linear kernel.

True classes →	Airplanes	Cars(rear)	Cars(side)	Faces	Motorbikes
Airplanes	**96.7**	0.2	0.6	2	3.4
Cars(rear)	0.4	**98.2**	1	1.1	2.4
Cars(side)	0.2	0	**97.6**	0.2	0.3
Faces	1	0.6	0.1	**94.2**	0.6
Motorbikes	1.8	1.1	0.8	2.4	**93.4**
Mean ranks	1.04	1.03	1.06	1.06	1.09

Table 4 shows the confusion matrix and the mean ranks[6] for SVM_1 with 10-fold cross-validation for comparison. The results obtained with 10-fold cross-validation outperform those obtained with 2-fold cross-validation. This is natural, as the number of training images increases.

5.0.4 10 Class Dataset.

For the experiments with our 10 class database we only used the independently built visual vocabulary. Therefore in the following we ignore the subscript 2 from SVM_2. Table 5.0.4 shows the confusion matrix and the mean ranks obtained for this dataset using a 5-fold cross-validation.

[6]These are the mean position of the correct labels when labels output by the multi-class classifier are sorted by the classifiers' scores.

Table 5: Confusion matrix and mean ranks for SVM.

classes	bikes	boats	books	cars	chairs	flowers	phones	r. signs	shoes	s. toys
bikes	**69.2**	1.7	1.9	1	5.3	1.2	0.4	3.2	1.2	1.3
boats	3.7	**79.3**	5.2	4.6	7.7	1.7	1.2	1.3	1.8	1.7
books	1.3	2.2	**70.3**	2.4	2.7	0.6	4.6	4.5	0.6	1.7
cars	4.2	3.7	2.4	**72.1**	8.3	0.3	3.1	3.1	1.5	0.8
chairs	10.8	3.9	5.3	5.4	**58.8**	2.2	2.7	5.8	1.3	1.1
flowers	1.2	1.2	1.3	1.6	1.2	**86.7**	1.6	1.6	0.7	0.8
phones	1.9	0.7	4.1	3.7	2.8	1.4	**70.4**	1.7	1.3	1.4
road signs	1.7	1.9	2.4	1.9	4.9	1.1	2.7	**69**	1.4	1.2
shoes	3.6	5.2	4.5	6.4	6.7	1.6	11.6	8.3	**86.3**	10.8
soft toys	2.4	0.2	2.6	0.9	1.6	3.2	1.7	1.5	3.9	**79.2**
mean ranks	1.5	1.3	1.5	1.3	1.9	1.3	1.9	1.7	1.3	1.2

5.0.5 Incorporating Geometry Information.

Table 6 shows the correct classification rates (M_{ii}) for each class obtained with the boosting approach without adding geometric information. The first row corresponds to the approach h_k where only single keypatch based weak classifiers were selected, and the second row shows results of the $h_{k,l}$ approach corresponding to weak classifiers based on pair of keypatches. In the third row we show the results of the SVM (the diagonal of the confusion matrix shown in Table 5.0.4) for comparison. All results were obtained by 5-fold cross-validation.

Table 6: Correct classification rates for: boosting without geometry ($h_k, h_{k,l}$); SVM with a linear kernel; boosting all types of weak classifiers h_{all} and boosting SVM with all types of weak classifiers (SVM_{all}). The standard error on the correct rate for each category is about 0.4%.

classes	bikes	boats	books	cars	chairs	flowers	phones	r. signs	shoes	s. toys	mean
h_k	61.7	74.5	67.0	55.6	50.7	82.5	67.6	61.4	73.9	68.9	**66.4**
$h_{k,l}$	64.6	76.1	68.5	61.0	50.7	84.6	69.6	64.8	76.6	69.2	**68.6**
SVM	69.2	79.3	70.3	72.1	58.8	86.7	70.4	69.0	86.3	79.2	**74.1**
h_{all}	70.0	73.8	68.2	64.1	57.4	82.9	68.0	61.9	75.2	76.2	**69.8**
SVM_{all}	74.6	81.8	78.2	77.5	65.2	89.6	76.0	76.2	83.8	83.8	**78.7**

We can see that $h_{k,l}$ outperforms h_k, but both boosting approaches have much lower performance than the linear SVM. We also tested the quadratic kernel for SVM as it implicitly considers keypatch pairs but the results were very similar to those of the linear kernel.

Furthermore, we investigated how well each weak classifier type performed when it was used exclusively for boosting. Results are given in Table 7 and selected weak classifier examples are shown in Figure 6.

We then combined the 17 types of geometric weak classifiers with hypotheses h_k and $h_{k,l}$. This (see fourth row of Table 6) slightly improved on the boosting results without geometry (first two rows) but gave still lower performance than the SVM.

Finally, we combined the SVM outputs with the weak classifiers using generalized AdaBoost. First the SVM outputs were normalized to $[-1, 1]$ using a sigmoid fit[7] [**?**]. This classifier was considered as first "weak" classifier h_1 of the boosting approach (see Eqn (1)) and the corresponding α_1 and $D^2(i)$ were accordingly computed (see Eqn's (3) and (2)). Other weak classifiers were selected from the full set of 19's h's. The second row of Table 7 shows how often each classifier type was used. Clearly h_B^{kl} and h_σ^{kl} are importnat complements to the SVM.

[7] This transformation of SVM outputs to confidence was also applied when we ranked the outputs from different classes.

Table 7: Mean correct rates when boosting individual weak classifier types (first row) and their percentage of being chosen when combined with SVM.

h_σ^k	h_σ^{kl}	$h_{\sigma=}^{kl}$	h_θ^k	h_θ^{kl}	$h_{\theta=}^{kl}$	$h_{\sigma\theta}^k$	$h_{\sigma\theta}^{kl}$	$h_{\sigma\theta=}^{kl}$	h_B^k	h_B^{kl}	$h_{k\cap l}$	$h_{k\in l}$	$h_{k\subset l}$	$h_{k\propto l}$	$h_{k\in\aleph_l^5}$	$h_{k\in\aleph_l^}$
63.6	66.5	46.2	62.1	62.6	48.8	61.6	64.5	48.8	62.8	63.8	63.3	64.1	53.8	58.5	62.4	64.5
2	21.2	2.8	0.4	13.5	3.2	0.4	9	1.6	3.8	35.7	2.7	0.8	0.1	0.3	0.9	1.6

Table 8: Confusion matrix and mean ranks for SVM_{all}.

classes	bikes	boats	books	cars	chairs	flowers	phones	r. signs	shoes	s. toys
bikes	**74.6**	1.6	1.5	0.6	5.5	1.2	0	1	0.6	0.8
boats	3.3	**81.8**	4.1	4.7	4.6	1.2	0.4	1	2.1	0.8
books	0	2.1	**78.2**	1.3	2.3	0	4.8	3.3	0.7	1.1
cars	3.8	3.7	2.2	**77.5**	7	0.4	2	3.3	1.3	0.4
chairs	10.4	2.8	4.4	4.8	**65.2**	2.1	2.8	4.3	2.5	0
flowers	1.2	0.9	0	0.6	1.8	**89.6**	0	1.4	0.6	0.8
phones	0.8	0.7	3	2.9	2.3	0.4	**76**	1	2.1	0.4
road signs	1.3	0.9	2.2	1.6	4.6	1.3	2.4	**76.2**	1.3	0
shoes	2.9	5.3	3.7	6	5.5	1.3	10.4	7.6	**83.8**	11.9
soft toys	0.4	0.2	0.7	0	1.2	2.5	1.2	0.9	5	**83.8**
mean ranks	1.4	1.3	1.4	1.3	1.8	1.2	1.9	1.5	1.5	1.1

The SVM performance was significantly improved (see Table 6 fifth row). Table 8 shows the confusion matrix and the mean ranks for this combined classifier.

6 Conclusions

In this chapter we have presented a simple approach to generic visual categorization using feature vectors constructed from clustered descriptors of image patches. This approach can easily handle variations in view and lighting. Furthermore it is robust to background clutter, occlusion as well as intra-class variations. It was tested on different datasets which showed the strength and the weaknesses of the method. Using an easy dataset (FPZ) we obtained excellent results both for object detection and for multi-class categorization. However our in-house dataset which is much more challenging showed that further improvements are necessary.

We explored the possibility of improving the accuracy using geometric information. In contrast to approaches such as [6, 11], where due to relatively strong geometrical (shape) constraints the method requires the alignment and segregation of different views of objects in the dataset, we proposed to incorporate geometric constraints as weak conditions. We defined and selected from a multitude of geometry based weak classifiers (several millions) and combine them effectively with the original SVM classifier using generalized AdaBoost. Results have been given on a challenging 10-class dataset which is publicly available. The benefits of the proposed method are its invariance and good accuracy. Overall improvement in error rate has been demonstrated through the use of geometric information, relative to results obtained in the absence of geometric information.

While we have explored 19 types (17 with geometry) of weak classifier, many more can be envisaged for future work. Geometric properties are of course widely used in matching. It will be interesting to explore how recent progress in this domain such as techniques in [7, 10] can be exploited for categorization. It will also be interesting to evaluate other approaches to boosting in the multi-class case such as

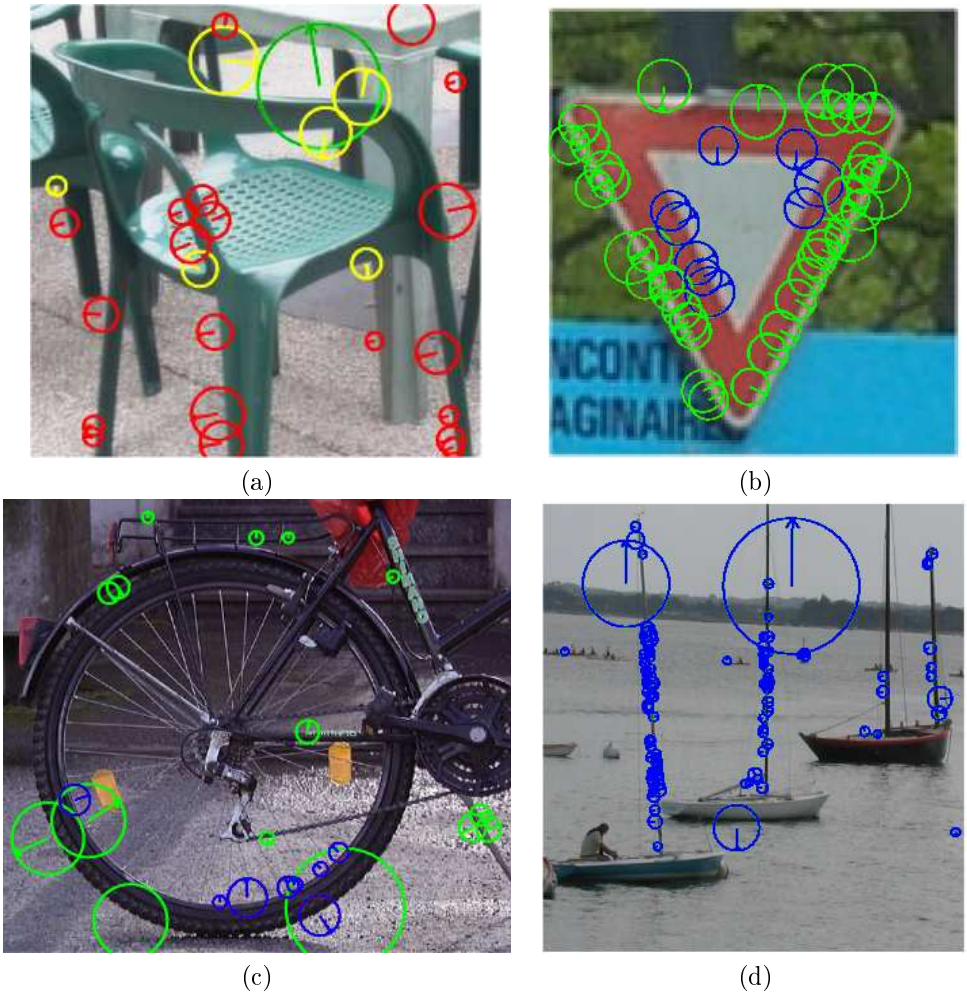

Figure 6: The three most relevant single keypatches h^k for the "chair" classifier are shown in (a). The following images show the single most relevant weak classifier based on pairs of patches for types (b) h^{kl}_{σ}, (c) $h_{k \subset l}$ and (d) $h_{k \cap l}$ respectively in the case of "road sign", "flowers" and "boat" classifiers. In each case we show *all* patches of type k (in blue) and *all* patches of type l (in green). Not all of these patches verify the respective geometric condition. For (d) $h_{k \cap l}$ it happends that the most relevant weak classifier $h_{k \subset l}$ for the "boat" classifier was obtained for $k = l$ hence only blue circles are shown.

the joint-boosting proposed in [24], which promises improved generalization performance and the need for fewer weak classifiers.

However, one of the main inconveniences of the proposed approach is the cost of the training which does not scale well with the number of images and number of classes (all weak classifiers must be tested on the whole training set at each step of the boosting). One way to reduce the search is to build a vocabulary of doublets (pair of keypatches) using only the most relevant visual words as in [21].

More recently, in [19] we have shown that approaches based on soft clustering using GMM rather than K-means can enable substantial improvements in accuracy. It was also shown that when combined with adaptation techniques drawn from speech recognition, such approaches can scale well with the number of classes. It will be an interesting challenge to incorporate geometric information with such soft clustering approaches.

7 Acknowledgments

This work was supported by the European Project IST-2001-34405 LAVA (Learning for Adaptable Visual Assistants, http://www.l-a-v-a.org). We are grateful to DARTY for their permission to acquire images in their shops, to INRIA for the use of their multi-scale interest point detector and to TU Graz for the bikes image database.

References

[1] A. Amir, J. Argillander, M. Berg, S.-F. Chang, M. Franz, W. Hsu, G. Iyengar, J. Kender, L. Kennedy, C.-Y. Lin, M. Naphade, A. Natsev, J. Smith, J. Tesic, G. Wu, R. Yang, and D. Zhang. IBM research TRECVID-2004 video retrieval system. In *Proc. of TREC Video Retrieval Evaluation*, 2004.

[2] P. Carbonetto, N. de Freitas, and K. Barnard. A statistical model for general contextual object recognition. In *Proc. ECCV*, volume 1, pages 350–362, 2004.

[3] Y. Chen and J. Z. Wang. Image categorization by learning and reasoning with regions. *JMLR*, 5:913–939, 2004.

[4] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Proc. ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[5] J. Farquhar, S. Szedmak, H. Meng, and J. Shawe-Taylor. Improving "bag-of-keypoints" image categorisation. Technical report, University of Southampton, 2005.

[6] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proc. CVPR*, volume 2, pages 264–271, 2003.

[7] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *Proc. ECCV*, volume 1, pages 40–54, 2004.

[8] W. H. Hsu and S.-F. Chang. Visual cue cluster construction via information bottleneck principle and kernel density estimation. In *Proc. CIVR*, 2005.

[9] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proc. ECML*, volume 1398, pages 137–142, 1998.

[10] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *Proc. BMVC*, volume 2, pages 959–968, 2004.

[11] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Proc. ECCV Workshop on Statistical Learning in Computer Vision*, pages 17–32, 2004.

[12] Y. Li, J. A. Bilmes, and L. G. Shapiro. Object class recognition using images of abstract regions. In *Proc. ICPR*, volume 1, pages 40–44, 2004.

[13] H. Lodhi, J. Shawe-Taylor, N. Christianini, and C. Watkins. Text classification using string kernels. In *Advances in Neural Information Processing Systems*, volume 13, 2001.

[14] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, pages 1150–1157, 1999.

[15] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proc. ECCV*, volume 1, pages 128–142, 2002.

[16] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *Proc. ECCV*, volume 2, pages 71–84, 2004.

[17] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu. GCap: Graph-based automatic image captioning. In *Proc. CVPR Workshop on Multimedia Data and Document Engineering*, 2004.

[18] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proc. ICML*, 2000.

[19] F. Perronnin, C. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. Submitted to ECCV 2006.

[20] R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[21] J. S. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. F. Feeman. Discovering objects and their localization in images. In *Proc. ICCV*, pages 370–377, 2005.

[22] J. S. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, volume 2, pages 1470–1477, 2003.

[23] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *Proc. ICML*, 2000.

[24] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: Efficient boosting procedures for multiclass object detection. In *Proc. CVPR*, volume 2, pages 762–769, 2004.

[25] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[26] L. Zhu, A. Rao, and A. Zhang. Theory of keyblock-based image retrieval. *ACM Transactions on Information Systems*, 20(2):224–257, 2002.