Visual Categorization with Bags of Keypoints

Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, Cédric Bray

Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan, France
{gcsurka,cdance}@xrce.xerox.com

Abstract. We present a novel method for generic visual categorization: the problem of identifying the object content of natural images while generalizing across variations inherent to the object class. This *bag of keypoints* method is based on vector quantization of affine invariant descriptors of image patches. We propose and compare two alternative implementations using different classifiers: Naïve Bayes and SVM. The main advantages of the method are that it is simple, computationally efficient and intrinsically invariant. We present results for simultaneously classifying seven semantic visual categories. These results clearly demonstrate that the method is robust to background clutter and produces good categorization accuracy even without exploiting geometric information.

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. To manage such collections it is useful to have access to high-level information about objects contained in the image. Given an appropriate categorization of image contents, one may efficiently search, recommend, react to or reason with new image instances.

We are thus confronted with the problem of *generic visual categorization*. We should like to identify processes that are sufficiently generic to cope with many object types simultaneously and which are readily extended to new object types. At the same time, these processes should handle the variations in view, imaging, lighting and occlusion, typical of the real world, as well as the intra-class variations typical of semantic classes of everyday objects.

The task-dependent and evolving nature of visual categories motivates an example-based machine learning approach. This paper presents a *bag of keypoints* approach to visual categorization. A bag of keypoints corresponds to a histogram of the number of occurrences of particular image patterns in a given image. The main advantages of the

method are its simplicity, its computational efficiency and its invariance to affine transformations, as well as occlusion, lighting and intra-class variations.

It is important to understand 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.

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. Most previous work on detection has centered on machine learning approaches to detecting faces, cars or pedestrians [1]-[6] 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 [5], neither of which is necessary in our approach.

Our *bag of keypoints* approach can be motivated by an analogy to learning methods using the *bag-of-words* representation for text categorization [7]-[10]. The idea of adapting text categorization approaches to visual categorization is not new. Zhu et al [11] 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. The idea of clustering invariant descriptors of image patches has previously been applied to the problem of texture classification [12]-[14]. Clearly the problem of texture classification is rather different from that of generic categorization. Therefore it is natural that these approaches differ from ours. While our method uses clustering to obtain quite high-dimensional feature vectors for a classifier, these texture recognizers use clustering to obtain relatively low-dimensional histograms and evaluate the similarity of these histograms to previously seen probability densities. In [12]-[13] filter responses are clustered and the recognition is done using the closest model measured by a χ^2 test. Lazebnik *et al* [14] cluster affine invariant interest points in each image individually and summarize the distribution of the descriptors in form of a signature composed of representative cluster members and weights proportional to cluster sizes. Signatures of different images are compared using the Earth Mover's Distance [15].

Recently Fergus *et al* [16] proposed a visual categorization method based on invariant descriptors of image patches. Their method exploits a probabilistic model that combines likelihoods for appearance, relative scale and position, as well as a model of

the statistics of their patch detector. This elegant approach has a number of limitations. Firstly the method is not efficient: even when models are restricted to 6 image patches and training images only contain up to 30 patches, days of CPU time are required to learn several categories. Secondly, views of objects used for training must be segregated, for instance into cars (rear) and cars (side). This is unsurprising given the use of an explicit 2D model of relative positions.

In section 2 we explain the categorization algorithms and the choice of their components. In section 3 we present results from applying of the algorithms to the dataset of Fergus *et al* and to a more challenging seven class dataset. We demonstrate that our approach is robust to the presence of background clutter and produces state-of-the-art recognition performance.

2. The method

The main steps of our method are:
- Detection and description of image patches
- Assigning patch descriptors to a set of predetermined clusters (a *vocabulary*) with a vector quantization algorithm
- Constructing a *bag of keypoints*, which counts the number of patches assigned to each cluster
- Applying a multi-class classifier, treating the bag of keypoints as the feature vector, and thus determine which category or categories to assign to the image.

Ideally these steps are designed to maximize classification accuracy while minimizing computational effort. Thus, the descriptors extracted in the first step should be invariant to variations that are irrelevant to the categorization task (image transformations, lighting variations and occlusions) but rich enough to carry enough information to be discriminative at the category level. The vocabulary used in the second step should be large enough to distinguish relevant changes in image parts, but not so large as to distinguish irrelevant variations such as noise.

We refer to the quantized feature vectors (cluster centres) as "keypoints" by analogy with "keywords" in text categorization. However, in our case "words" do not necessarily have a repeatable meaning such as "eyes", or "car wheels", nor is there an obvious best choice of vocabulary. Rather, our goal is to use a vocabulary that allows good categorization performance on a given training dataset. Therefore the steps involved in training the system allow consideration of multiple possible vocabularies:
- Detection and description of image patches for a set of labeled training images
- Constructing a set of vocabularies: each is a set of cluster centres, with respect to which descriptors are vector quantized.
- Extracting bags of keypoints for these vocabularies
- Training multi-class classifiers using the bags of keypoints as feature vectors

- Selecting the vocabulary and classifier giving the best overall classification accuracy.

We now discuss the choices made for each step in more detail.

2.1 Feature extraction

In computer vision, local descriptors (i.e. features computed over limited spatial support) have proved well-adapted to matching and recognition tasks, as they are robust to partial visibility and clutter. Such tasks require descriptors that are repeatable. Here, we mean repeatable in the sense that if there is a transformation between two instances of an object, corresponding points are detected and (ideally) identical descriptor values are obtained around each. This has motivated the development of several scale and affine invariant point detectors, as well as descriptors that are resistant to geometric and illumination variations [17]-[21].

It was shown in [21] that if we have affine transformation between two images a scale invariant point detector is not sufficient to have the stability of the point's location. Therefore we preferred to work with the *Harris affine detector* described in [21]. However, the reader should be aware that the benefits of this choice are not clear cut: firstly because most real world objects have three-dimensional structures whose variations are not well captured by affine transformations; secondly since attempts to increase the invariance of a feature typically result in a loss of discriminative information.

Harris affine points are detected by an iterative process. Firstly, positions and scales of interest points are determined as local maxima (in position) of a scale-adapted Harris function, and as local extrema in scale of the Laplacian operator. Then an elliptical (i.e. affine) neighborhood is determined. This has a size given by the selected scale and a shape given by the eigenvalues of the image's second moment matrix. The selection of position/scale and the elliptical neighborhood estimation are then iterated and the point is kept only if the process converges within a fixed number of iterations.

The affine region is then mapped to a circular region, so normalizing it for affine transformations. Scale Invariant Feature Transform (SIFT) descriptors [18] are computed on that region. SIFT descriptors are multi-image representations of an image neighborhood. They are Gaussian derivatives computed at 8 orientation planes over a 4x4 grid of spatial locations, giving a 128-dimension vector. Fig. 1 shows an example of the maps of gradient magnitude corresponding to the 8 orientations.

Fig. 1. (*From left to right*) A Harris affine region; the normalized region; and the 8 maps of gradient magnitude constituting the SIFT descriptor.

We prefer SIFT descriptors to alternatives such as steered Gaussian derivatives or differential invariants of the local jet for the following reasons:

1. They are simple linear Gaussian derivatives. Hence we expect them to be more stable to typical image perturbations such as noise than higher Gaussian derivatives or differential invariants.

2. The use of a simple Euclidean metric in the feature space seems justified. In the case of differential invariants obtained by a combination of the components of the local jet, the use of a Mahalanobis distance is more appropriate. For instance, one would expect a second derivative feature to have a higher variance than a first derivative. Selecting an appropriate Mahalanobis distance a priori seems challenging. It would not be appropriate to use the covariance matrix of SIFT descriptors over the entire dataset, since this is predominantly influenced by inter-class variations (or more precisely, by variations between keypoints that we do not wish to ignore). Measuring a Mahalanobis distance would probably requiring manual specification of multiple homologous matching points between different images of objects of the same category, seriously working against our objective of producing a simple and automated categorization system.

3. There are far more components to these feature vectors (128 rather than 12 to 16). Hence we have a richer and potentially more discriminative representation.

Recently Mikolajczyk *et al* [22] have compared several descriptors for matching and found that SIFT descriptors perform best.

2.3 Visual vocabulary construction

In our method, the vocabulary is a way of constructing a feature vector for classification that relates "new" descriptors in query images to descriptors previously seen in training. One extreme of this approach would be to compare each query descriptor to all training descriptors: this seems impractical given the huge number of training descriptors involved (over 600 000 for our data set). Another extreme would be to try to identify a small number of large "clusters" that are good at discriminating a given class: for instance [16] operates with 6 parts per category. In practice we find that the best tradeoffs of accuracy and computational efficiency are obtained for intermediate sizes of clustering.

Most clustering or vector quantization algorithms are based on iterative square-error partitioning or on hierarchical techniques. Square-error partitioning algorithms attempt to obtain the partition which minimizes the within-cluster scatter or maximizes the between-cluster scatter. Hierarchical techniques organize data in a nested sequence of groups which can be displayed in the form of a dendrogram or a tree. They need

some heuristics to form clusters and hence are less frequently used than square-error partitioning techniques in pattern recognition.

We chose to use the simplest square-error partitioning method: k-means [23]. This algorithm proceeds by iterated assignments of points to their closest cluster centers and recomputation of the cluster centers. Two difficulties are that the k-means algorithm converges only to local optima of the squared distortion, and that it does not determine the parameter k. There exist methods allowing to automatically estimating the number of clusters. For example, Pelleg et al [24] use cluster splitting to do it, where the splitting decision is done by computing 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 run k-means several times with different number of desired representative vectors (k) and different sets of initial cluster centers. We select the final clustering giving the lowest empirical risk in categorization [25].

2.3 Categorization

Once descriptors have been assigned to clusters to form feature vectors, we reduce the problem of generic visual categorization to that of multi-class supervised learning, with as many classes as defined visual categories. The categorizer performs two separate steps in order to predict the classes of unlabeled images: training and testing. During training, labeled data is sent to the classifier and used to adapt a statistical decision procedure for distinguishing categories. Among many available classifiers, we compared the Naïve Bayes classifier for its simplicity and its speed, and the Support Vector Machine since is it often known to produce state-of-the-art results in high-dimensional problems.

2.3.2 Categorization by Naïve Bayes

Naïve Bayes [26] is a simple classifier used often in text categorization. It can be viewed as the maximum *a posteriori* probability classifier for a generative model in which: 1) a document category is selected according to class prior probabilities; 2) each word in the document is chosen *independently* from a multinomial distribution over words specific to that class. While independence is a *naïve* assumption, the accuracy of the Naïve Bayes classification is typically high [27].

Now, considering visual categorization, assume we have a set of labeled images $I=\{\mathbf{I_i}\}$ and a vocabulary $V=\{\mathbf{v_i}\}$ of representative keypoints (i.e. cluster centers). Each descriptor extracted from an image is labeled with the keypoint to which it lies closest in feature space. We count the number $N(t,i)$ of times keypoint $\mathbf{v_i}$ occurs in image $\mathbf{I_i}$. To categorize a new image, we apply Bayes's rule and take the largest *a posteriori* score as the prediction:

$$P(C_j \mid I_i) \ \alpha \ \ P(C_j)P(I_i \mid C_j) = P(C_j)\prod_{t=1}^{|V|} P(v_t \mid C_j)^{N(t,i)}. \tag{1}$$

It is evident in this formula that Naïve Bayes requires estimates of the class-conditional probabilities of keypoint v_t given category C_j. In order to avoid probabilities of zero, these estimates are computed with Laplace smoothing:

$$P(v_t \mid C_j) = \frac{1 + \sum_{\{I_i \in C_j\}} N(t,i)}{|V| + \sum_{s=1}^{|V|} \sum_{\{I_i \in C_j\}} N(s,i)}. \tag{2}$$

2.3.2 Categorization by SVM

The SVM classifier finds a hyperplane which separates two-class data with maximal *margin* [25]. The margin is defined as the distance of the closest training point to the separating hyperplane. For given observations $\mathbf{X}$, and corresponding labels $\mathbf{Y}$ which takes values ± 1, one finds a classification function:

$$\mathbf{f}(\mathbf{x}) = \mathbf{sign}(\mathbf{w}^T\mathbf{x} + \mathbf{b}) \tag{3}$$

where $\mathbf{w}, \mathbf{b}$ represents the parameters of the hyperplane.

Data sets are not always linearly separable. The SVM takes two approaches to this problem. Firstly it introduces an error weighting constant *C* which penalizes misclassification of samples in proportion to their distance from the classification boundary. Secondly a mapping Φ is made from the original data space of $\mathbf{X}$ to another feature space. This second feature space may have a high or even infinite dimension. One of the advantages of the SVM is that it can be formulated entirely in terms of scalar products in the second feature space, by introducing the *kernel*

$$\mathbf{K}(\mathbf{u}, \mathbf{v}) = \Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) \tag{4}$$

Both the kernel $\mathbf{K}$ and penalty *C* are problem dependent and need to be determined by the user.

In the kernel formulation, the decision function can be expressed as

$$\mathbf{f}(\mathbf{x}) = \mathbf{sign}(\sum_i \mathbf{y}_i \alpha_i \mathbf{K}(\mathbf{x}, \mathbf{x}_i) + \mathbf{b}). \tag{5}$$

where $\mathbf{x}_i$ are the training features from data space X and $\mathbf{y}_i$ is the label of $\mathbf{x}_i$. Here the parameters α_i are typically zero for most *i*. Equivalently, the sum can be taken only over a select few of the $\mathbf{x}_i$. These feature vectors are known as *support vectors*. It can be shown that the support vectors are those feature vectors lying nearest

to the separating hyperplane. In our case, the input features $\mathbf{x_i}$ are the binned histograms formed by the number of occurrences of each keypoint $\mathbf{v_i}$ from the vocabulary V in the image $\mathbf{I_i}$.

In order to apply the SVM to multi-class problems we take the one-against-all approach. Given an m-class problem, we train m SVM's, each 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 assign it to the class with the largest SVM output.

3   Experiments

We present results from three experiments. In the first we explore the impact of the number of clusters on classifier accuracy and evaluate the performance of the Naïve Bayes classifier. We then explore the performance of the SVM on the same problem. The first two experiments were conducted on an in-house seven class dataset. In the last experiment we describe results on the four class dataset employed in [16].

Our in-house database contains 1776 images in seven classes[1]: faces, buildings, trees, cars, phones, bikes and books. Fig. 2 shows some examples from this dataset.
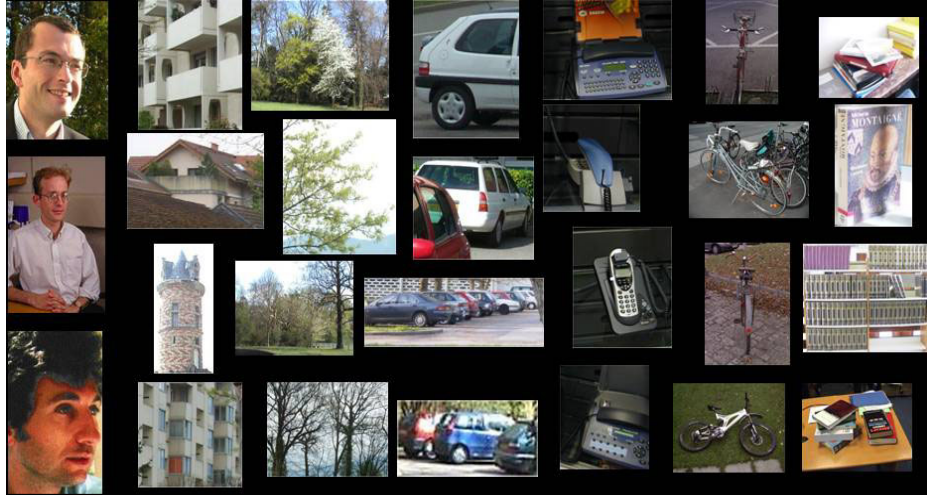

Fig. 2. Example images from our in-house database

It is a challenging dataset, not only because of the large number of classes, but also because it contains images with highly variable poses and significant amounts of

---

[1] We hope to make this dataset publicly available soon. It contains 792 faces, 150 buildings, 150 trees, 201 cars, 216 phones, 125 bikes and 142 books.

background clutter, sometimes presence of objects from multiple classes although a large proportion of each image's area is occupied by the target category. The images have resolutions between 0.3 and 2.1 megapixels and were acquired with a diverse set of cameras. The images are color but only the luminance component is used in our method. These were gathered by XRCE and Graz University, excepting faces which were downloaded from the Web.

We used three performance measures to evaluate our multi-class classifiers.

– The confusion matrix:

$$M_{ij} = \frac{\left|\left\{\mathbf{I_k} \in \mathbf{C_j} : \mathbf{h(I_k)} = \mathbf{i}\right\}\right|}{\left|\mathbf{C_j}\right|} \qquad (6)$$

where $\mathbf{i, j} \in \left\{\mathbf{1, ..., N_c}\right\}$, $\mathbf{C_j}$ is the set of test images from category $\mathbf{j}$ and $\mathbf{h(I_k)}$ is the category which obtained the highest classifier output for image $\mathbf{I_k}$.

– The overall error rate:

$$R = \mathbf{1} - \frac{\sum_{\mathbf{j=1}}^{\mathbf{N_c}} \left|\mathbf{C_j}\right| M_{jj}}{\sum_{\mathbf{j=1}}^{\mathbf{N_c}} \left|\mathbf{C_j}\right|} \qquad (7)$$

– The mean ranks. These are the mean position of the correct labels when labels output by the multi-class classifier are sorted by the classifier score.

Each performance metric was evaluated with 10-fold cross validation.

3.2 Naïve Bayes Results

In Fig. 3 we present the overall error rates using Naïve Bayes as a function of the number of clusters k. Each point in Fig. 3 is the best of 10 random trials of k-means. The standard-error on the maximum is in the range [1, 3]%.[2] The error rate only improves slightly as we move from $k = 1000$ to $k = 2500$. We therefore assert that $k = 1000$ present a good trade-off between accuracy and speed[3].

---

[2] That is, if the experiment of taking the maximum of 10 is repeated multiple times with different random clusterings, the standard deviations of the result for the different classes are in the range [1,3]%.

[3] It takes about one minute to get predicted labels on the whole database using the Naive Bayes classifier with $k = 1000$ on a 2 GHz processor.
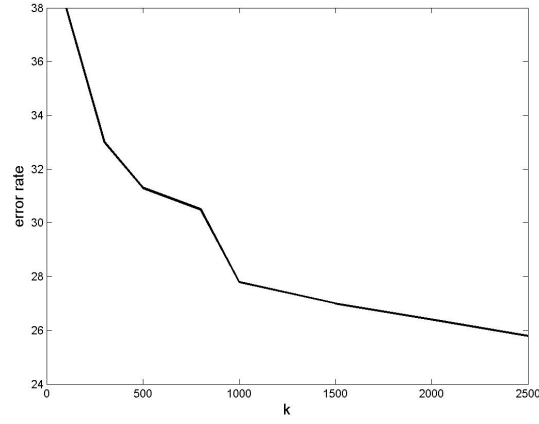
Fig. 3. The lowest overall error rate (percentage) found for different choices of *k*

Table 1 shows the performance as a function of category obtained with this *k*.

Table 1. Confusion matrix and the mean rank for the best vocabulary (*k=1000*).

True classes →	*faces*	*buildings*	*trees*	*cars*	*phones*	*bikes*	*books*
faces	**76**	4	2	3	4	4	13
buildings	2	**44**	5	0	5	1	3
trees	3	2	**80**	0	0	5	0
cars	4	1	0	**75**	3	1	4
phones	9	15	1	16	**70**	14	11
bikes	2	15	12	0	8	**73**	0
books	4	19	0	6	7	2	**69**
Mean ranks	1.49	1.88	1.33	1.33	1.63	1.57	1.57

In the clustering process, there is a risk of bias since images from different categories contain different numbers of interest points. We therefore used random samples of the training data in the clustering step, each sample containing 5000 interest points randomly chosen from each class (there are around 640 000 interest points extracted from 1776 training images).

Fig. 4 illustrates example clusters obtained for the selected "best" vocabulary.
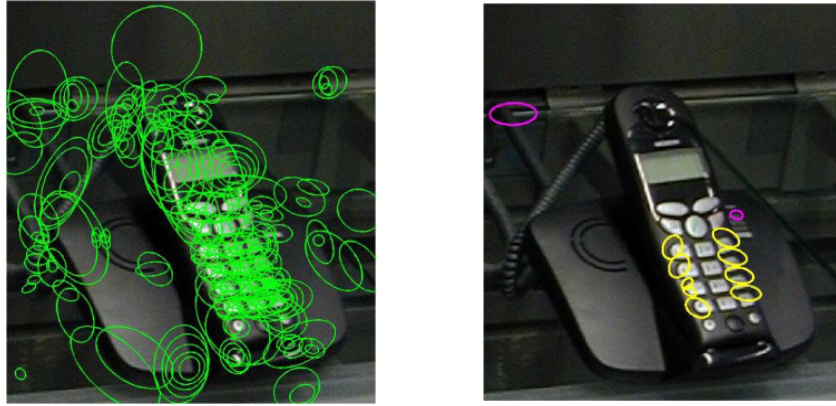
Fig. 4. *Left* all patches detected for this image. *Right* patches from two selected clusters occurring in this image (yellow and magenta ellipses).

The algorithm handles easily multiple objects in the same images (see Fig. 5), occlusion, partial view, any orientation (Fig. 6).

Fig. 5. Images correctly classified containing multiple objects of the same category.

Fig. 6. Profile face, partial view of a car, roof of a house correctly classified as face, cars, building.

In Fig. 7 we show examples where images were well classified even if most of the detected interest points were on the "background".
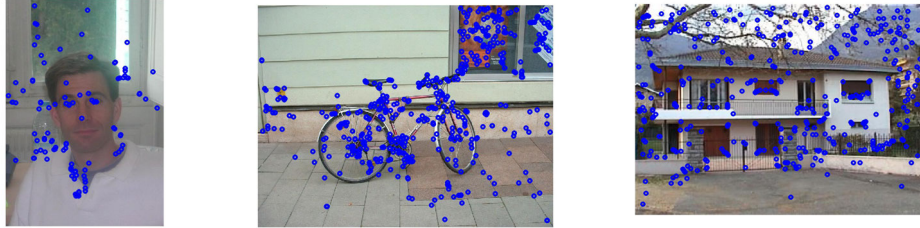
Fig. 7. Images where background clutter is in a higher percentage than interest points on the object.

Fig. 8 shows some images from the database where objects from other categories were also present in the image and the first 3 ranked categories for each of them. These images were not considered as multiple labels but labeled by the main object in the image.

| phones, books, cars | bikes, buildings, cars | buildings, cars, faces |

Fig. 8. Images where multiple objects were present and the first three ranked labels.

Finally, Fig. 9 shows some false alarm with the first label and the true label with its rank.

| face(1)… book(2) | trees(1)…face(5) | phones(1)…cars(2) |

Fig. 9. Examples of incorrectly ranked images. The correct label's rank is also shown.

3.2 SVM Results

Results from applying the SVM are given in Table 2.

Table 2. Confusion matrix and mean rank for SVM (*k=1000, linear kernel*).

True classes →	faces	buildings	trees	cars	phones	bikes	books
faces	**98**	14	10	10	34	0	13
buildings	1	**63**	3	0	3	1	6
trees	1	10	**81**	1	0	6	0
cars	0	1	1	**85**	5	0	5
phones	0	5	4	3	**55**	2	3
bikes	0	4	1	0	1	**91**	0
books	0	3	0	1	2	0	**73**
Mean ranks	1.04	1.77	1.28	1.30	1.83	1.09	1.39

As expected the SVM outperformed Naïve Bayes, reducing the overall error rate from 28 to 15%. We also obtained better mean ranks except in the case of cars. The 2% error rate for faces seems to be comparable with state-of-the-art approaches, and is perhaps surprising that our method exploits virtually no geometric information. However a direct comparison is not possible since our method must also cope with background images from the other classes. This low error rate comes at a price of increased confusion of other categories with faces because of the larger number of faces in the training set.

In training this SVM we used the same best vocabulary with *k=1000* as for Naïve Bayes. We compared linear, quadratic and cubic SVM's and found that linear method gave the best performance (except in the case of cars where a quadratic SVM gave better results). The parameter *C* was determined for each SVM and values of around *C*=0.005 typically gave the best results.

3.3   Results on database [16]

We also tested our approach with the freely available[4], database used in [16]; i.e. the following five object classes and the set of background images: faces (450 images), airplanes (side) (1074 images), cars (rear) (651 images), cars (side) (720 images), and motorbikes (side) (826 images).

The results in Table 3 shows, that the images in this database are much easier to classify than our database. These results cannot be directly compared with the results stated in [16]. This is because their correct classification rate is a result of classifying images of a given class against background images (a two-class problem), which is different from our case where we conduct a full multi-class comparison using ranks (we hope to publish direct comparison soon). However, the diagonal elements of Table 3 suggest that our approach will give at least as good results as theirs in two-class

---

[4] Downloaded from http://www.robots.ox.ac.uk/~vgg/data except for cars (side) that comes from http://l2rcs.cs.uiuc.edu/~cogcomp/index_research.html.

classification (generally the two-class classification is more robust than the multi-class one). While the confusion matrices given in Table 2 of [16] are not directly comparable with our results, they suggest a rather high confusion rate in the multi-class case.

Table 3. .Confusion matrix and mean rank for SVM (*k=1000, linear kernel*).

True classes →	faces (frontal)	airplanes (side)	cars (rear)	cars (side)	motorbikes (side)
faces(frontal)	**94**	0.4	0.7	0	1.4
airplanes (side)	1.5	**96.3**	0.2	0.1	2.7
cars (rear)	1.9	0.5	**97.7**	0	0.9
cars(side)	1.7	1.9	0.5	**99.6**	2.3
motorbikes (side)	0.9	0.9	0.9	0.3	**92.7**
Mean ranks	1.07	1.04	1.03	1.01	1.09

4   Conclusion

We have presented a simple but novel approach to generic visual categorization using feature vectors constructed from clustered descriptors of image patches. This approach has been evaluated on a seven category database, demonstrating the method is robust to background clutter and produces good categorization accuracy even without exploiting geometric information. Our results with SVM are clearly superior to those obtained with the simple Naïve Bayes classifier. To our best knowledge this is the largest number of visual categories that has ever been subjected to simultaneous experiment.

Much future work remains. As we extend to more visual categories, the discriminative power of the appearance of unordered image patches will not suffice and we will need to extend the categorizer to incorporate geometric information. We will also need to extend our method for cases where the object of interest occupies a small fraction of the field of view, and to investigate many promising alternatives for each step of the basic method (point detection, clustering and classification).

References

[1]   E. Osuna, R. Freund, F and Girosi. Training support vector machines: An application to face detection, CVPR (Computer Vision and Pattern Recognition), 1997.

[2]   C. Papageorgiou, T. Evgeniou and T. Poggio. A trainable pedestrian detection system, IEEE Conference on Intelligent Vehicles, 1998.

[3]   H. Schneiderman and T. Kanade, "A Statistical method for 3D object detection applied to faces and cars", CVPR, 2000.

[4]   P. Viola and M. Jones, Rapid object detection using a boosted cascade of simple features, CVPR, 2001

[5]   S.Z. Li, L. Zhu, Z.Q. Zhang, A. Blake, H.J. Zhang and H. Shum, Statistical learning of multi-view face detection, ECCV (European Conference on Computer Vision), 2002.

[6]   R. Ronfard, C. Schmid, and B. Triggs, Learning to parse pictures of people, ECCV, 2002.

[7]   T. Joachims. Text categorization with support vector machines: Learning with many relevant features, ECML, 1998.

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

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

[10] N. Cristianini, J.Shawe-Taylor and H. Lodhi, Latent Semantic Kernels, *Journal of Intelligent Information Systems*, **18** (2), 127-152, 2002.

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

[12] Th. Leung and J. Malik, Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons, ICCV 1999.

[13] M. Varma and A. Zisserman, Classifying materials from images: to cluster or not to cluster?, ECCV 2002.

[14] S. Lazebnik, C. Schmid and J. Ponce, Sparse texture representation using affine-invariant neighborhoods, CVPR 2003.

[15] Y. Rubner and C. Tomasi, Texture-based Image Retrieval Without Segmentation, ICCV 1999.

[16] R. Fergus, P.Perona, and A. Zisserman, Object Class Recognition by Unsupervised Scale-Invariant Learning, CVPR 2003.

[17] T. Lindenberg, Scale-space theory in computer vision, Kluwer Academic Publishers, 1994.

[18] D. G. Lowe, Object Recognition from local scale–invariant features, ICCV (International Conference on Computer Vision), 1999.

[19] J. Matas, J. Burianek, and J. Kittler. Object recognition using the invariant pixel-set signature, BMVC (British Machine Vision Conference), 2000.

[20] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo, ICCV, 2001.

[21] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector, ECCV, 2002.

[22] K. Mikolajczyk and C. Schmid, A performance evaluation of local descriptors, CVPR, 2003.

[23] O. Duda, P.E. Hart, D.G. Stork, Pattern classification, John Wiley & Sons, 2000.

[24] D. Pelleg and A. Moore. X-Means: Extending K-means with Efficient Estimation of the Number of Clusters, International Conference on Machine Learning, 2000.

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

[26] D. D. Lewis, Naïve Bayes at forty: The independence assumption in information retrieval, ECML, 1998.

[27] P. Domingos and M. Pazzani, On the optimality of simple Bayesian classifier under zero-one loss, *Machine Learning,* **29**, 1997.