Beyond Sliding Windows: Object Localization by *Efficient Subwindow Search*

Christoph H. Lampert Matthew B. Blaschko Thomas Hofmann
Max Planck Institute for Biological Cybernetics Google Inc.
72076 Tübingen, Germany Zurich, Switzerland
{chl,blaschko}@tuebingen.mpg.de

Abstract

Most successful object recognition systems rely on binary classification, deciding only if an object is present or not, but not providing information on the actual object location. To perform localization, one can take a sliding window approach, but this strongly increases the computational cost, because the classifier function has to be evaluated over a large set of candidate subwindows.

In this paper, we propose a simple yet powerful branch-and-bound scheme that allows efficient maximization of a large class of classifier functions over all possible subimages. It converges to a globally optimal solution typically in sublinear time. We show how our method is applicable to different object detection and retrieval scenarios. The achieved speedup allows the use of classifiers for localization that formerly were considered too slow for this task, such as SVMs with a spatial pyramid kernel or nearest neighbor classifiers based on the χ^2-distance. We demonstrate state-of-the-art performance of the resulting systems on the UIUC Cars dataset, the PASCAL VOC 2006 dataset and in the PASCAL VOC 2007 competition.

1. Introduction

Recent years have seen great progress in the area of object category recognition for natural images. Recognition rates beyond 95% are the rule rather than the exception on many datasets. However, in their basic form, many state-of-the-art methods only solve a binary classification problem. They can decide whether an object is present in an image or not, but not where exactly in the image the object is located.

Object localization is an important task for the automatic understanding of images as well, *e.g*. to separate an object from the background, or to analyze the spatial relations of different objects in an image to each other. To add this functionality to generic object categorization systems, sliding window approaches have established themselves as state-of-the-art. Most successful localization techniques at the recent PASCAL VOC 2007 challenge on object category localization relied on this technique. The sliding window principle treats localization as localized detection, applying a classifier function subsequently to subimages within an image and taking the maximum of the classification score as indication for the presence of an object in this region. However, already an image of as low resolution as 320×240 contains more than *one billion* rectangular subimages. In general, the number of subimages grows as n^4 for images of size $n \times n$, which makes it computationally too expensive to evaluate the quality function exhaustively for all of these. Instead, one typically uses heuristics to speed up the search, which introduces the risk of mispredicting the location of an object or even missing it.

In this paper, we propose a method to perform object localization in a way that does not suffer from these drawbacks. It relies on a branch-and-bound scheme to find the global optimum of the quality function over all possible subimages, thus returning the same object locations that an exhaustive sliding window approach would. At the same time it requires much fewer classifier evaluations than there are candidate regions in the image—often even less than there are pixels— and typically runs in linear time or faster.

The details of this method, which we call *Efficient Subwindow Search (ESS)*, are explained in Section 2. ESS allows object localization by localized detection and also localized image retrieval for classifiers which previously were considered unusable in these applications, because they were too slow or showed too many local maxima in their classification scores. We will describe the proposed systems in Sections 3–5, demonstrating their state-of-the-art performance. First, we give an overview of other approaches for object localization and their relation to ESS.

1.1. Sliding Window Object Localization

Many different definitions of object localization exist in the literature. Typically, they differ in the form that the location of an object in the image is represented, *e.g*. by its center point, its contour, a bounding box, or by a pixel-wise segmentation. In the following we will only study localization where the target is to find a bounding box around the

object. This is a reasonable compromise between the simplicity of the parametrization and its expressive power for subsequent scene understanding. An additional advantage is that it is much easier to provide ground truth annotation for bounding boxes than *e.g.* for pixel-wise segmentations.

In the field of object localization with bounding boxes, sliding window approaches have been the method of choice for many years [3, 6, 7, 11, 19]. They rely on evaluating a quality function f, *e.g.* a classifier score, over many rectangular subregions of the image and taking its maximum as the object's location. Formally, we write this as

$$R_{obj} = \operatorname{argmax}_{R \subseteq I} \; f(R), \tag{1}$$

where R ranges over all rectangular regions in the image I.

Because the number of rectangles in an $n \times n$ image is of the order n^4, this maximization usually cannot be done exhaustively. Instead, several heuristics have been proposed to speed up the search. Typically, these consist of reducing the number of necessary function evaluations by searching only over a coarse grid of possible rectangle locations and by allowing only rectangles of certain fixed sizes as candidates [7, 11, 19]. Alternatively, local optimization methods can be applied instead of global ones, by first identify promising regions in the image and then maximizing f by a discrete gradient ascent procedure from there [3, 6].

The reduced search techniques sacrifice localization robustness to achieve acceptable speed. Their implicit assumption is that the quality function is smooth and slowly varying. This can lead to false estimations or even complete misses of the objects locations, in particular if the classifier function's maximum takes the form of a sharp peak in the parameter space. However, such a sharply peaked maximum is exactly what one would hope for to achieve accurate and reliable object localization.

2. Efficient Subwindow Search (ESS)

In contrast to approximation methods, ESS is guaranteed to find the globally maximal region, independent of the shape of f's quality landscape. At the same time ESS is very fast, because it relies on a branch-and-bound search instead of an exhaustive search. This speed advantage allows the use of more complex and better classifiers.

ESS also differs from previous branch-and-bound approaches for object localization, because it is flexible in the choice of quality function. Previous methods were restricted to either finding simple parametric objects like lines and circles in line drawings [4], or to nearest-neighbor classification using a fixed L^2-like distance between the features points in an image to sets of rigid prototypes [13]. To our knowledge, ESS is currently the only efficient method that allows globally optimal localization of arbitrary objects in images with results equivalent to an exhaustive sliding windows search.

2.1. Branch-and-Bound Search

The underlying intuition of ESS is the following: even though there is a very large number of candidate regions for the presence of the objects we are searching for, only very few of them can actually contain object instances. It is wasteful to evaluate the quality function for all candidate regions if only the value of the best few is required. Instead, one should target the search directly to identify the regions of highest score, and ignore the rest of the search space where possible.

The *branch-and-bound* framework allows such a targeted search. It hierarchically splits the parameter space into disjoint subsets, while keeping bounds of the maximal quality on all of the subsets. This way, large parts of the parameter space can be discarded early during the search process by noticing that their upper bounds are lower than a guaranteed score from some previously examined state.

For ESS, the parameter space is the set of all possible rectangles in an image, and subsets are formed by imposing restrictions on the values that the rectangle coordinates can take. We parameterizes rectangles by their top, bottom, left and right coordinates (t, b, l, r). We incorporate uncertainty in these values by using intervals instead of single integers for each coordinate. This allows the efficient representation of set of rectangles as tuples $[T, B, L, R]$, where $T = [t_{low}, t_{high}]$ etc., see Figure 1 for an illustration.

For each rectangle set, we calculate a bound for the highest score that the quality function f could take on any of the rectangles in the set. ESS terminates when it has identified a rectangle with a quality score that is at least as good as the upper bound of all remaining candidate regions. This guarantees that a global maximum has been found. If tight enough bounds are available, ESS typically converges to a globally optimal solution much faster than the worst case complexity indicates. In our experiments, the speed was at most $\mathcal{O}(n^2)$ for $n \times n$ images instead of $\mathcal{O}(n^4)$.

ESS organizes the search over candidate sets in a best-first manner, always examining the rectangle set that looks most promising in terms of its quality bound. The candidate set is split along its largest coordinate interval into halves, thus forming two smaller disjoint candidate sets. The search is stopped if the most promising set contains only a single rectangle with the guarantee that this is the rectangle of globally maximal score. Algorithm 1 gives pseudo-code for ESS using a priority queue to hold the search states.

Two extensions of the basic search scheme provide additional functionality: to favor boxes with specific shapes properties one can add a geometric penalization term to f, *e.g.* a Gaussian that takes its maximum at a certain rectangle size or aspect ratio. Of course this term has to be taken in account when bounding the values of f.

To find multiple object locations in an image, the best-first search can be performed repeatedly. Whenever an ob-

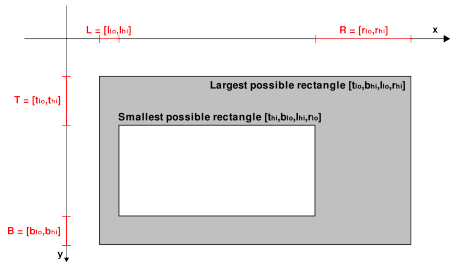

Figure 1. Representation of rectangle sets by 4 integer intervals.

ject is found, the corresponding region is removed from the image and the search is restarted until the desired number of locations have been returned. In contrast to simply continuing the search after the best location has been identified, this avoids the need for a non-maximum suppression step as is usually required in sliding window approaches.

2.2. Bounding the Quality Function

To use ESS for a given quality function f, we require a function $\hat{f}$ that bounds the values of f over sets of rectangles. Denoting rectangles by R and sets of rectangles by $\mathcal{R}$, the bound has to fulfill the following two conditions:

$$i) \quad \hat{f}(\mathcal{R}) \geq \max_{R \in \mathcal{R}} f(R),$$

$$ii) \quad \hat{f}(\mathcal{R}) = f(R), \quad \text{if } R \text{ is the only element in } \mathcal{R}.$$

Condition $i)$ ensures that $\hat{f}$ acts as an upper bound to f, whereas condition $ii)$ guarantees the optimality of the solution to which the algorithm converges.

Note that for any f there is a spectrum of possible bounds $\hat{f}$. On the one end, one could perform an exhaustive search to achieve exact equality in (i). On the other end, one could set $\hat{f}$ to a large constant on everything but single rectangles. A good bound $\hat{f}$ is located between these extremes, fast to evaluate but also tight enough to ensure fast convergence. In the following sections we show how such bounding functions $\hat{f}$ can be constructed for different choices of f.

3. Application I: Localization of non-rigid objects using a bag of visual words kernel

We begin by demonstrating ESS in the situation of generic object class localization. We make use of a *bag of visual words (bovw)* image representation: for each image in a given set of training images $I^1, \ldots, I^N$, we extract local image descriptors such as SIFT [17]. The resulting descriptors are vector quantized using a K-entry codebook of visual word prototypes. As result, we obtain keypoint locations x_j^i with discrete cluster indices $c_j^i \in \{1, \ldots, K\}$.

We represent images or regions within images by their cluster histograms, *i.e.* by histograms that count how many

Algorithm 1 Efficient Subwindow Search

Require: image $I \in \mathbb{R}^{n \times m}$
Require: quality bounding function $\hat{f}$ (see text)
Ensure: $(t_{\max}, b_{\max}, l_{\max}, r_{\max}) = \operatorname{argmax}_{R \subset I} f(R)$
  initialize P as empty priority queue
  set $[T, B, L, R] = [0, n] \times [0, n] \times [0, m] \times [0, m]$
  repeat
    split $[T, B, L, R] \rightarrow [T_1, B_1, L_1, R_1] \dot{\cup} [T_2, B_2, L_2, R_2]$
    push $([T_1, B_1, L_1, R_1],\ \hat{f}([T_1, B_1, L_1, R_1]))$ into P
    push $([T_2, B_2, L_2, R_2],\ \hat{f}([T_2, B_2, L_2, R_2]))$ into P
    retrieve top state $[T, B, L, R]$ from P
  until $[T, B, L, R]$ consists of only one rectangle
  set $(t_{\max}, b_{\max}, l_{\max}, r_{\max}) = [T, B, L, R]$

feature points of each cluster index occur. The histograms of the training images are used to train a support-vector machine (SVM) [20]. To classify whether a new image I contains an object or not, we build its cluster histogram h and decide based on the value of the SVM decision function. Despite the simplicity, variants of this method have proven very successful for object classification in recent years [2, 8, 16, 21, 24].

3.1. Construction of a Quality Bound

To perform localization, we first assume a linear kernel over the histograms. In its canonical form, the corresponding SVM decision function is $f(I) = \beta + \sum_i \alpha_i \langle h, h^i \rangle$, where $\langle ., . \rangle$ denotes the scalar product in $\mathbb{R}^K$. h^i are the histograms of the training examples and α_i and β are the weight vectors and bias that were learned during SVM training. Because of the linearity of the scalar product, we can rewrite this expression as a sum over per-point contribution with weights $w_j = \sum_i \alpha_i h_j^i$:

$$f(I) = \beta + \sum_{j=1}^{n} w_{c_j}. \tag{2}$$

Here c_j is the cluster index belonging to the feature point x_j and n is the total number of feature points in I. This form allows one to evaluate f over subimages $R \subset I$ by summing only over the feature points that lie within R. When we are only interested in the argmax of f over all $R \subset I$ –as in Equation (1)– we can drop the bias term β.

It is now straightforward to construct a function $\hat{f}$ that bounds f over sets of rectangles: set $f = f^+ + f^-$, where f^+ contains only the positive summands of Equation (2) and f^- only the negative ones. If we denote by $R_{\max}$ the largest rectangle and by $R_{\min}$ the smallest rectangle contained in a parameter region $\mathcal{R}$, then

$$\hat{f}(\mathcal{R}) := f^+(R_{\max}) + f^-(R_{\min}) \tag{3}$$

has the desired properties $i)$ and $ii)$. At the same time, our parametrization of the rectangle sets allows efficient calcu-

lation of $R_{\max}$ and $R_{\min}$ from the minima and maxima of the individual coordinate intervals (see Figure 1). Using integral images we can make the evaluations of f^+ and f^- $\mathcal{O}(1)$ operations, thus making the evaluation of $\hat{f}$ a constant time operation. That the evaluation time of $\hat{f}$ is *independent of the number of rectangles* contained in $\mathcal{R}$ is a crucial factor in why ESS is fast.

3.2. Experiments

The *bovw* representation disregards all spatial relations between feature points in an image. This total invariance to changes in the object geometry, the pose and the viewpoint make the *bovw* classifier especially eligible for the detection of object classes that show a large amount of variance in their visual appearance as is the case, *e.g.*, for many animals. The case where we can make use of geometric information such as a predominant pose will be treated in Section 4.

3.2.1 PASCAL VOC 2006 dataset

In a first set of experiments, we tested the *bovw* based localization on the *cat* and *dog* categories of the publicly available PASCAL VOC 2006 dataset[1]. It consists of 5,304 realistic images containing 9,507 object from 10 categories in total. The dataset is split into a *trainval* part, on which all algorithm development is performed, and a *test* part that is reserved for the final evaluation.

For the *bovw* representation we extract SURF features [2] from keypoint locations and from a regular grid and quantize them using a 1000 entry codebook that was created by K-means clustering a random subset of 50,000 descriptors. As positive training examples for the SVM we use the ground truth bounding boxes that are provided with the dataset. As negative examples we sample box regions from images with negative class label and from locations outside of the object region in positively labeled images.

To measure the system's performance on a pure localization task, we apply ESS to only the test images that actually contain objects to be localized (*i.e.* cats or dogs). For each image we return the best object location and evaluate the result by the usual VOC method of scoring: a detected bounding box is counted as a correct match if its area overlap with the corresponding ground truth box is at least 50%. To each detection a confidence score is assigned that we set to the value of the quality function on the whole image. Figure 2 contains *precision–recall* plots of the results. The curves' rightmost points correspond to returning exactly one object per image. At this point of operation, approximately 55% of all cats bounding boxes returned are correct and 47% of all dog boxes. At the same time, we correctly localize 50% of all cats in the dataset and 42% of all dogs. Moving along the curve to the left, only objects are included into the evaluation which have higher confidence scores assigned to them. This generally improves the localization precision.

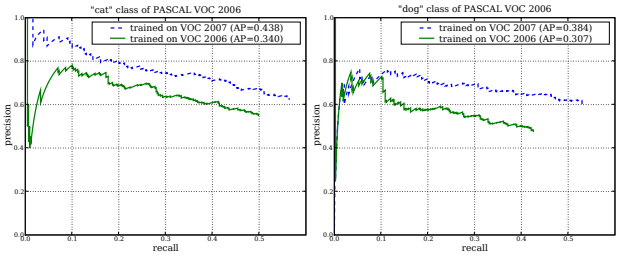

Figure 2. Recall–Precision curves of ESS *bovw* localization for classes *cat* and *dog* of the VOC 2006 dataset. Training was performed either on VOC 2006 (solid line) or VOC 2007 (dashed).

method \ data set	cat	dog
ESS w/ bag-of-visual-words kernel	0.223	0.148
Viitaniemi/Laaksonen [23]	0.179	0.131
Shotton/Winn [9]	0.151	0.118

Table 1. Average Precision (AP) scores on the PASCAL VOC 2006 dataset. ESS outperforms the best previously published results.

As no other results on pure localization on the PASCAL VOC datasets have been published so far, we also performed the more common evaluation scenario of combined localization and retrieval. For this, the method is applied to all images of the test set, no matter if they contain the object to be searched for or not. It is the task of the algorithm to avoid false positives *e.g.* by assigning them a low confidence score. The performance is measured using the evaluation software provided in the PASCAL VOC challenges: from the *precision–recall* curves, the *average precision (AP)* measure is calculated, which is the average of the maximal precision within different intervals of recall, see [9] for details. Table 1 contains the results, showing that ESS improves over the best results that have been achieved in the VOC 2006 competition or in later publications. Note that the *AP* values in Table 1 are not comparable to the ones in Figure 2, since the experiments use different test sets.

3.2.2 PASCAL VOC 2007 challenge

An even larger and more challenging dataset than PASCAL VOC 2006 is the recently released VOC 2007[2]. It consists of 9,963 images with 24,640 object instances. We trained a system analogous to the one described above, now using the 2007 training and validation set, and let the system participate in the PASCAL VOC challenge 2007 on multi-view object localization. In this challenge, the participants did

[1] www.pascal-network.org/challenges/VOC/voc2006/

[2] www.pascal-network.org/challenges/VOC/voc2007/

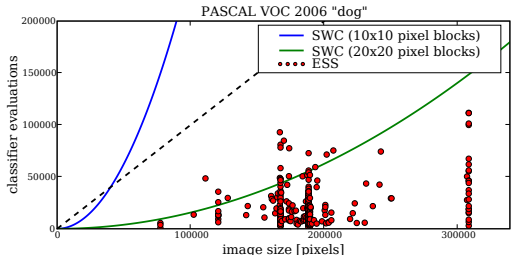

Figure 3. Speed of ESS compared to a sliding window classifier (SWC). The number of classifier evaluations required by ESS on the VOC 2006 *dog* images (red dots) are plotted against the image size. SWC needs to be restricted to a grid of 20×20 pixel cells (green line) to achieve performance comparable with ESS. 10×10 pixel cells are not enough (blue line).

not have access to the ground truth of the test data, but had to submit their localization results, which were then evaluated by the organizers. This form of evaluation allows the comparison different methods on a fair basis, making it less likely that the algorithms are tuned to the specific datasets.

With AP scores of 0.240 for cats and 0.162 for dogs, ESS clearly outperformed the other participants on these classes, with the runner-up scores being 0.132 for cats and 0.126 for dogs. By adopting a better image-based ranking algorithm, we were able improve the results to 0.331 and 0.177 respectively. The latter results are part of the ongoing second round of the competition.

In addition, we used the system that had been trained on the 2007 *trainval* data, and evaluated its performance on the 2006 *test* set. The results are included in Figure 2. The combination achieves higher recall and precision than the one trained on the 2006 data, showing that ESS with a bag-of-visual-words kernel generalizes well and is able to make positive use of the larger number of training images available in the 2007 dataset.

3.3. Localization Speed

Since we optimize the same quality function as a sliding window approach, the same scores could have been achieved by exhaustively evaluating the quality function over all rectangles in the images. However, this is only a theoretical possibility, since it would be much too slow in practice. With the majority of images being sized between 500×333 and 640×480 pixels, an exhaustive sliding window localizer would on average require over 10 billion classifier evaluations per image. In contrast to this, ESS on average required less than 20,000 evaluations of the quality bound per image, less than 0.1 evaluations per pixel. This resulted in an average search time of below $40ms$ per image on a 2.4 GHz PC. Building a sliding window classifier of comparable speed would require reducing the spatial accuracy to blocks of at least 20×20 pixels, see Figure 3.

4. Application II: Localization of rigid objects using a Spatial Pyramid Kernel

For rigid and typically man-made object classes like cars or buildings, better representations exist than the *bag-of-visual-words* used in the previous section. In particular *hierarchical spatial pyramids* of features have recently proven very successful. These are formed by overlaying the image with rectangular grids of different sizes and calculating *bovw* histograms for each of the grid cells, see *e.g.* [14] for the exact construction.

Spatial pyramids have successfully been used for localization, but they were restricted to a small number of pyramid levels (typically 2 or 3). Additionally, heuristic pruning techniques were necessary to keep the complexity at an acceptable level [6]. In the following we show that ESS overcomes this limitation and allows efficient localization with pyramids as fine-grained as 10 levels and more without the risk of missing promising object locations.

4.1. Classification with a Spatial Pyramid Kernel

We make use of an SVM classifier with linear kernel on hierarchical spatial pyramid histograms. The decision function f on a new image I is calculated as

$$f(I) = \beta + \sum_{l=1}^{L} \sum_{\substack{i=1,\dots l \\ j=1,\dots,l}} \sum_{k=1}^{N} \alpha_k^{l,(i,j)} \langle h_{l,(i,j)}, h^k_{l,(i,j)} \rangle, \quad (4)$$

where $h_{l,(i,j)}$ is the histogram of all features of the image I that fall into the spatial grid cell with index (i,j) of an $l \times l$ spatial pyramid. $\alpha_k^{l,(i,j)}$ are the coefficients learned by the SVM when trained with training histograms $h^k_{l,(i,j)}$.

Using the linearity of the scalar products, we can transform this into a sum of per-point contributions and evaluate it on subimages:

$$f(R) = \beta + \sum\nolimits_{m=1}^{n} \sum_{l=1}^{L} \sum_{\substack{i=1,\dots l \\ j=1,\dots,l}} w_{c_m}^{l,(i,j)}, \quad (5)$$

where $w_c^{l,(i,j)} = \sum_k \alpha_k^{l,(i,j)} h^k_{l,(i,j);c}$, if the feature point x_m has cluster label c and falls into the (i,j)-th cell of the l-th pyramid level of R. Otherwise, we set $w_c^{l,(i,j)} = 0$. As before, we can ignore the bias term β for the maximization.

A comparison with Equation (2) shows that Equation (5) is a sum of *bovw* contributions, one for each level and cell index (l,i,j). We bound each of these as explained in the previous section: for a given rectangle set $\mathcal{R}$, we calculate the largest and the smallest possible extent that a grid cell $R^{(l,i,j)}$ can have. Calling these $R^{(l,i,j)}_{max}$ and $R^{(l,i,j)}_{min}$, we obtain an upper bound for the cell's contribution by adding all weights of the feature points with positive weights $w_c^{l,(i,j)}$

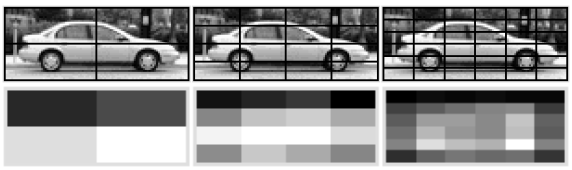

Figure 4. Spatial Pyramid Weights. Top row: Example of a training image with its pyramid sectors for levels 2, 4 and 6. Bottom row: the energy of corresponding pyramid sector weights as learned by the SVM (normalized per level). Feature points in brighter regions in general have higher discriminative power.

that fall into $R_{max}^{(l,i,j)}$ and the weight of all feature points with negative weights that fall into $R_{min}^{(l,i,j)}$. An upper bound for f is obtained by summing the bounds for all levels and cells. If we make use of two integral images per triplet (l,i,j), evaluating $f(R)$ becomes an $\mathcal{O}(1)$ operation. This shows that also for the spatial pyramid representation, an efficient branch-and-bound search is possible.

4.2. UIUC Car dataset

We test our setup on the UIUC Car database[3], which is an example of a dataset with rigid object images (cars) from a single viewpoint. In total there are 1050 training images of fixed size 100×40 pixels. 550 of these show a car in side-view, the rest shows other scenes or parts of objects. There are two test sets of images with varying resolution. The first consists of 170 images containing 200 cars from a side view of size 100×40. The other test set consists of 107 images containing 139 cars in sizes between 89×36 and 212×85. We use the dataset in its original setup [1] where the task is pure localization. Ground truth annotation and evaluation software is provided by the creators of the dataset.

4.3. Experiments

From the UIUC Car training images, we extract SURF descriptors at different scales on a dense pixel grid and quantize them using a 1000 entry codebook that was generated from 50,000 randomly sampled descriptors. Since the training images already either exactly show a car or not at all, we do not require additional bounding box information and train the SVM with hierarchical spatial pyramid kernel on the full training images. We vary the number of pyramid levels between $L = 1$ (*i.e.* a *bovw* without pyramid structure) and $L = 10$. The most fine-grain pyramid therefore uses all grids from 1×1 to 10×10, resulting in a total of 385 local histograms. Figure 4 shows an example image from training set and the learned classifier weights from different pyramid levels, visualized by their total energy over the histogram bins. On the coarser levels, more weight is assigned to the lower half of the car region than to the upper half. On the finer pyramid levels, regions of specific characteristics form, *e.g.* the wheels becomes very discriminative whereas the top row and the bottom corners can safely be ignored.

[3] http://l2r.cs.uiuc.edu/cogcomp/Data/Car/

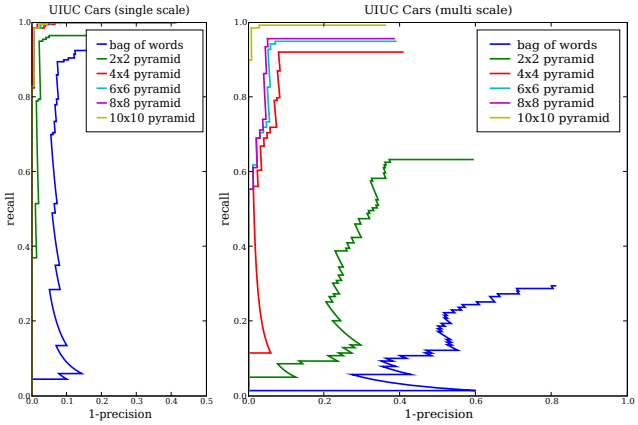

Figure 5. Results on UIUC Cars Dataset (best viewed in color): $1 - precision$ vs $recall$ curves for bag-of-features and different size spatial pyramids. The curves for single-scale detection (left) become nearly identical when the number of levels increases to 4×4 or higher. For the multi scale detection the curves do not saturate even up to a 10×10 grid.

method \data set	single scale	multi scale
ESS w/ 10×10 pyramid	1.5 %	1.4 %
ESS w/ 4×4 pyramid	1.5 %	7.9 %
ESS w/ bag-of-visual-words	10.0 %	71.2 %
Agarwal et al. [1]	23.5 %	60.4 %
Fergus et al. [10]	11.5 %	—
Leibe et al. [15]	2.5 %	5.0%
Fritz et al. [12]	11.4 %	12.2%
Mutch/Lowe [18]	0.04 %	9.4%

Table 2. Error rates on UIUC Cars dataset at the point of equal precision and recall.

At test time, we search for the best three car subimages in every test image as described in Section 2.1, and for each detection we use its quality score as confidence value. As it is common for the UIUC Car dataset, we evaluate the system's performance by a $1 - precision$ vs. $recall$ curve. Figure 5 shows the curves for several different pyramid levels. Table 2 contains error rates at the point where precision equals recall, comparing the results of ESS with the currently best published results. Note that the same dataset has also been used in many other setups, *e.g.* using different training sets or evaluation methods. Since the results of these are not comparable, we do not include them.

The table shows that localization with a flat *bovw*-kernel works acceptably for the single scale test set but poorly for multi scale. Using ESS with a finer spatial grid improves the error rates strongly, up to the level where the method

clearly outperforms all previously published approaches on the multi scale dataset and all but one on the single scale dataset.

Note that for the single scale test set, a direct sliding window approach with fixed window size 100×40, would be computationally feasible as well. However, there is no advantage of this over ESS, as the latter requires even fewer classifier evaluations on average, and at the same time allows the application of the same learned model to the multi-scale situation without retraining.

5. Application III: Image Part Retrieval using a χ^2-Distance Measure

ESS can be applied in more areas than only object localization. In the following, we give a short description how to use ESS for image part retrieval *i.e*. to retrieve images from a database based on queries that only have to match a part of the target image. This allows one not only to search for objects or persons, but also *e.g*. to find trademarked symbols on Internet image collections or in video archives.

5.1. χ^2-kernel for image similarity

We adopt a *query-by-example* framework similar to [22], where the query is a part of an image, and we are interested in all frames or scenes in a video that contain the same object. For this, we use ESS to do a complete nearest-neighbor comparison between the query and all boxes in all database images. In contrast to previous approaches, this allows the system to rely on arbitrary similarity measures between regions, not just on the number of co-occurring features. In our example, we choose the χ^2-distance that has shown good performance for histogram-based retrieval and classification tasks [5].

To treat the retrieval problem in an optimization framework, we first define the localized similarity between a query region with *bovw*-histogram h^Q and an image I as

$$locsim(I, Q) = \max_{R \subseteq I} -\chi^2(h^Q, h^R) \tag{6}$$

where h^R is the histogram for the subimage R and

$$\chi^2(h^Q, h^R) = \sum_{k=1}^{K} \frac{(h_k^Q - h_k^R)^2}{h_k^Q + h_k^R}. \tag{7}$$

The retrieval task is now to identify the N images with highest localized similarity to Q as well as the region within each of them that best matches the query.

Since $locsim$ consists of a maximization over all subregions in an image, we can use ESS to calculate it. To construct the required bound, we first notice that the value of each histogram bin over a set of rectangles $\mathcal{R}$ can be bounded from below and from above by the number of keypoints with corresponding cluster index that fall into $R_{\min}$ and $R_{\max}$ respectively. We denote these bounds by $\overline{h}_k^R$ and $\underline{h}_k^R$. Each summand in (7) can now be bounded from below by

$$\frac{(h_k^Q - h_k^R)^2}{h_k^Q + h_k^R} \geq \begin{cases} (h_k^Q - \underline{h}_k^R)^2/(h_k^Q + \underline{h}_k^R) \text{ for } h_k^Q < \underline{h}_k^R, \\ 0 \qquad\qquad\qquad\qquad \text{for } \underline{h}_k^R \leq h_k^Q \leq \overline{h}_k^R, \\ (h_k^Q - \overline{h}_k^R)^2/(h_k^Q + \underline{h}_k^R) \text{ for } h_k^Q > \overline{h}_k^R, \end{cases}$$

and their negative sum bounds $-\chi^2(h^Q, h^R)$ from above.

5.2. Simultaneous ESS for multiple images

A direct application of ESS would consist of searching for the best region in each image and afterwards ranking them according to their score, from which we can determine the N highest. However, we can achieve a much more efficient search by combining the maximization over the N-best scores and the maximization over the regions R into a single best-first search. This is done by adding the start states of all images into the priority queue before starting the search. During the run, the candidate regions of all images are simultaneously brought into an order according to how relevant they are to the query. Search states from images that do not contain promising candidate regions always stay at the bottom of the queue and might never be expanded. Whenever a best match has been found, the states corresponding to the image found are removed from the search queue and the search is continued until N regions have been detected. In our experiments the combined search caused a 40 to 70-times speedup compared to the sequential search when N was set to approximately 1% of the total number of images in the database.

5.3. Experiments

We show the performance of ESS in localized retrieval by applying it to 10143 keyframes of the full-feature movie *"Ferris Bueller's Day Off"*. Each frame is 880×416 pixels large. For a given query, multi-image ESS is used to return the 100 images containing the most similar regions. Figure 6 shows a query region and some search results. Since keyframes within the same scene tend to look very similar, we show only one image per scene. ESS is able to reliably identify the *Red Wings* logo in different scenes regardless of strong background variations with only 4 falsely reported frames out of 100. The first error occurred at position 88. The search required less than 2s per returned image, scaling linearly in the number of output images and effectively sublinearly in the number of images in the database[4].

[4] Since every image has to be inserted into the search queue, the method cannot be sublinear in the sense of computation complexity. However, the observed growth of runtimes is sublinear, and the more images are added, the fewer operations per image are necessary on average to find the top N.

(a) Red Wings logo used as query. (b) Top results of local search

Figure 6. Image retrieval using a local χ^2 distance: the Red Wings logo (left) is used as a query region. b) shows the top results (one image per scene). The center image in the bottom row contains a false positive. The other images are correct detections.

6. Conclusion

We have demonstrated how to perform fast object localization and localized retrieval with results equivalent to an exhaustive evaluation of a quality function over all rectangular regions in an image down to single pixel resolution. Sliding window approaches have the same goal, but in practice they have to resort to subsampling techniques and approximations to achieve a reasonable speed. In contrast to this, our method retains global optimally in its search, which guarantees that no maxima of the quality function are missed or misplaced.

The gain in speed and robustness allows the use of better local classifiers (*e.g.* SVM with spatial pyramid kernel, nearest neighbor with χ^2-distance), for which we demonstrated excellent results on the UIUC Cars, the PASCAL VOC 2006 dataset and in the VOC 2007 challenge. We also showed how to integrate additional properties, *e.g.* shape penalties, and how to search over large image collections in sublinear time.

In future work, we plan to study the applicability of ESS to further kernel-based classifiers. We are also working on extensions to other parametric shapes, like groups of boxes, circles and ellipses. These are often more desirable in applications of biological, medical or industrial machine vision, where high speed and performance guarantees are important quality factors as well.

Acknowledgments

This work was funded in part by the EC project CLASS, IST 027978. The second author is supported by a Marie Curie fellowship under the EC project PerAct, EST 504321.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to Detect Objects in Images via a Sparse, Part-Based Representation. *PAMI*, 26(11):1475–1490, 2004.

[2] H. Bay, T. Tuytelaars, and L. J. Van Gool. SURF: Speeded Up Robust Features. In *ECCV*, pages 404–417, 2006.

[3] A. Bosch, A. Zisserman, and X. Muñoz. Representing Shape with a Spatial Pyramid Kernel. In *CIVR*, pages 401–408, 2007.

[4] T. M. Breuel. Fast Recognition using Adaptive Subdivisions of Transformation Space. In *CVPR*, pages 445–451, 1992.

[5] O. Chapelle, P. Haffner, and V. N. Vapnik. Support Vector Machines for Histogram-based Image Classification. *Neural Networks*, 10(5):1055–1064, 1999.

[6] O. Chum and A. Zisserman. An Exemplar Model for Learning Object Classes. In *CVPR*, 2007.

[7] N. Dalal and B. Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, pages 886–893, 2005.

[8] G. Dorko and C. Schmid. Selection of Scale-Invariant Parts for Object Class Recognition. *ICCV*, pages 634–640, 2003.

[9] M. Everingham, A. Zisserman, C. Williams, and L. V. Gool. The Pascal Visual Object Classes Challenge 2006 Results. Technical report, PASCAL Network, 2006.

[10] R. Fergus, P. Perona, and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *CVPR*, pages 264–271, 2003.

[11] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of Adjacent Contour Segments for Object Detection. *PAMI*, 30:36–51, 2008.

[12] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating Representative and Discriminative Models for Object Category Detection. In *ICCV*, pages 1363–1370, 2005.

[13] D. Keysers, T. Deselaers, and T. M. Breuel. Optimal Geometric Matching for Patch-Based Object Detection. *ELCVIA*, 6(1):44–54, 2007.

[14] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *CVPR*, pages 2169–2178, 2006.

[15] B. Leibe, A. Leonardis, and B. Schiele. Robust Object Detection with Interleaved Categorization and Segmentation. *IJCV Special Issue on Learning for Recognition and Recognition for Learning*, 2007 (in press).

[16] F.-F. Li and P. Perona. A Bayesian Hierarchical Model for Learning Natural Scene Categories. In *CVPR*, pages 524–531, 2005.

[17] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[18] J. Mutch and D. G. Lowe. Multiclass Object Recognition with Sparse, Localized Features. In *CVPR*, pages 11–18, 2006.

[19] H. A. Rowley, S. Baluja, and T. Kanade. Human Face Detection in Visual Scenes. In *NIPS 8*, pages 875–881, 1996.

[20] B. Schölkopf and A. Smola. *Learning With Kernels*. MIT Press, 2002.

[21] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering Objects and their Localization in Images. In *ICCV*, pages 370–377, 2005.

[22] J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *ICCV*, pages 1470–1477, 2003.

[23] V. Viitaniemi and J. Laaksonen. Techniques for Still Image Scene Classification and Object Detection. In *ICANN (2)*, volume 4132, pages 35–44, 2006.

[24] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study. *IJCV*, 73(2):213–238, 2007.