Generating object segmentation proposals using global and local search

Pekka Rantalankila, Juho Kannala, Esa Rahtu
University of Oulu

Abstract

We present a method for generating object segmentation proposals from groups of superpixels. The goal is to propose accurate segmentations for all objects of an image. The proposed object hypotheses can be used as input to object detection systems and thereby improve efficiency by replacing exhaustive search. The segmentations are generated in a class-independent manner and therefore the computational cost of the approach is independent of the number of object classes. Our approach combines both global and local search in the space of sets of superpixels. The local search is implemented by greedily merging adjacent pairs of superpixels to build a bottom-up segmentation hierarchy. The regions from such a hierarchy directly provide a part of our region proposals. The global search provides the other part by performing a set of graph cut segmentations on a superpixel graph obtained from an intermediate level of the hierarchy. The parameters of the graph cut problems are learnt in such a manner that they provide complementary sets of regions. Experiments with Pascal VOC images show that we reach state-of-the-art with greatly reduced computational cost.

1. Introduction

Automatic detection and recognition of objects from images are tasks which have been under active research during the recent years. The research efforts have led to significant improvements in the state-of-the-art in the field. For example, since the launch of the Pascal Visual Object Classes (VOC) challenge [6] the progress of the field has been regularly evaluated using standardized databases. Recently also other databases have been introduced, like the ImageNet[1] and the SUN dataset[2] , which contain more images and object classes than the VOC dataset. In fact, there is a trend towards larger and richer datasets and this sets up additional challenges for the detection systems.

A dominant paradigm in object detection and recognition has been to use so called bag-of-visual-words features or histograms of oriented gradients in a sliding window framework [17, 7]. Typically these approaches represent the visual appearance of a rectangular image window using histogram features, which encode information of texture, color or intensity gradients. The histogram features are then used as input to a discriminant function of a binary classifier for joint detection and recognition. Usually a separate classifier is trained for each object category which is to be recognized. Since the location, size and aspect ratio of the bounding boxes of objects are initially unknown, both feature extraction and evaluation of classifiers are performed for a very large number of densely placed image windows and hence the term *sliding window* is used.

[1] http://www.image-net.org/
[2] http://groups.csail.mit.edu/vision/SUN/

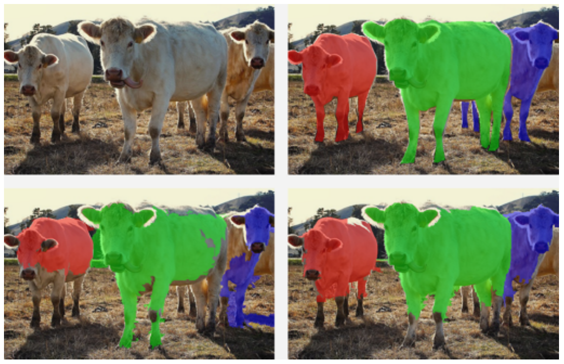

Figure 1. Example segmentations. Top: original image and ground truth. Bottom: colors indicate the best segmentations from pools of proposals generated by [16] (left) and our method (right).

However, despite its success, the sliding window technique suffers from the problem of high computational cost when the number of object categories is large or the applied classifiers are costly to evaluate. Therefore, several approaches have been proposed in order to address efficiency issues. For example, multi-stage cascade architectures have been proposed in which the early stages filter out a large portion of bad bounding box hypotheses using simple classifiers, which are cheaper to evaluate than the more accurate ones [17]. This implies that costly classifiers can be evaluated with a smaller set of windows. Still, a huge number of windows per image must be classified by some means and therefore the method is not suitable for large datasets with many object classes.

In addition, there are approaches which try to reduce the number of processed windows by sampling the space of bounding boxes non-uniformly and by discarding non-object windows in a category-independent manner [2, 14].

That is, the bounding box hypotheses which most likely contain background stuff (e.g. sky or water) and which are not likely to contain instances from *any* object category are discarded from further consideration using computationally cheap features and classifiers. Importantly, the cost of such stage is independent of the number of object categories which are searched for. Still, even for relative small images, such as those in Pascal VOC dataset, tens or hundreds of thousands of windows must be classified, at least once, or otherwise the level of attainable recall drops significantly. Hence, these approaches are not suitable for very large datasets although their performance scales better with respect to the number of object classes than that of the conventional approaches.

Further, the efficient subwindow search method proposed by [11] avoids explicit processing of all possible subwindows by using a branch and bound strategy. Nevertheless, this approach is not as generic as the conventional sliding window method but is limited to use relatively simple classifiers, which typically do not provide as good performance as their more complex alternatives. Therefore many of the recent approaches that produce state-of-the-art results use the sliding window approach or its variants [17, 7].

Recently, a promising research direction for object detection has emerged with approaches that generate object segmentation proposals [16, 4, 5]. Some of these approaches are very slow [4, 5], and hence not at all suitable for efficient and scalable frameworks. However, the approach of [16] is relatively efficient and was used as a first stage in object detection systems that won the detection challenges of ImageNet 2011 and Pascal VOC 2012.

In this paper, we describe a fast method for producing object segmentation proposals by grouping superpixels. The method can be used as a first stage in a class-generic object detection framework to limit the number of image regions passed to further processing stages, which may be class-specific. Our approach is inspired by [16] but provides considerably better segmentations as also shown in Fig. 1. Our contributions and their connection to closely related previous work are described in the next section.

2. Related Work

The closest works to ours are [16, 5, 4]. We take some of the best innovations from these previous works, which represent the current state-of-the-art, and propose several important developments that allow us to find an excellent balance between the computational efficiency and the number and coverage of segmented regions. We will show that our approach is approximately as fast as [16] but provides more accurate segmentations which better cover the objects in an image. Further, our approach is much faster than [5, 4] but provides segmentations of comparable accuracy.

The approach of [16] obtains the proposed object regions from a segmentation hierarchy which is formed by starting from an over-segmentation, which consists of regions (i.e. *superpixels*) obtained by [8], and then greedily merging pairs of adjacent regions until there is only one region left (i.e. the whole image is a single region). A single segmentation hierarchy is not enough to get a sufficiently diverse set of regions but by using several over-segmentations computed in several color spaces [16] is able to obtain reasonable levels of recall.

Our approach utilizes similar segmentation hierarchies as [16] but in a different manner. Instead of simply increasing the number of hierarchies by choosing several starting points from different over-segmentations, we branch the bottom-up segmentation hierarchy at a certain level into several branches each of which leads to a small set of binary foreground-background segmentations. That is, at a fixed level of the hierarchy we form a graph of this level's regions and obtain several binary segmentations by optimizing various versions of a cost function with graph cuts using different parameter settings and different subsets of regions as foreground and background seeds. In addition to this we take all the regions from the full hierarchy obtained by running the greedy merging process [16] until the end. Our approach is motivated by the observation that the greedy local search for pairs of adjacent regions works well in the beginning when sufficiently similar regions are merged (i.e. close to the bottom of the hierarchy) whereas "grabcut" [15] type of global segmentation allows efficient diversification of proposal regions when it is performed on a relatively rough segmentation level where the number of regions is not too high (i.e. the number of nodes in the graph is small).

The main difference to [16] is that besides the greedy merging process, which can be seen as a local search in the space of sets of superpixels, we also use global search which is realized by solving several graph cut problems on a superpixel graph. Further, for measuring the similarity of regions we use different color and texture features than [16], and, instead of using several segmentations obtained by running [8] with various parameter settings in different color spaces, we compute our initial over-segmentations by using the SLIC segmentation method [1] in addition to [8].

Binary foreground-background segmentations implemented by graph cuts have been used in [5, 4]. However, our work deviates from these previous works in two important aspects. Firstly, our approach is much faster than [5] and [4] since we use more efficient image features and are able to run graph cut based segmentation on coarser graphs. Secondly, we use training data to learn a threshold which determines the initial over-segmentation used as a starting point for the global search by graph cuts (i.e. the branching level in our segmentation hierarchy) and we also learn a ranking for the segmentation branches. That is, we learn a (partial) ordering for the seeding strategies and parameters of the graph cuts optimization and this allows us to add branches gradually in such a manner that the regions from newly added branches generally tend to give the highest possible increase in recall.

Hence, our approach is in contrast to [5, 4] which first generate a larger pool of proposal regions and then rank them according to their ”objectness”. If the aim is to apply the segmentation proposals as input to a detection cascade, such as in [16] or [12], and the later processing stages can process only a certain number of regions per image, it is computationally more efficient to directly generate an approximately correct number of good proposal regions than to first generate a larger set of regions, then rank the regions and select the desired number of top-ranked regions. Still, if a full ordering of the proposal regions would be necessary, our regions could be ranked using the techniques presented in [5] or [4], and also [14] or [18] if bounding boxes are used instead of the original regions.

Further, it should be noted that we solve graph cut optimization problems on a superpixel graph whereas [4] performs graph cut optimization on the original pixel graph. In our case the graph has less nodes which is beneficial (i.e. about hundreds of nodes instead of hundreds of thousands). Thus, in our approach the global search stage greatly benefits from the greedy local search which generates the initial segmentation for the global stage. Indeed, unlike previous approaches, we are able to utilize the observation that the greedy merging performs particularly well in the early stages of the bottom-up segmentation hierarchy when there usually are many similar superpixels.

3. Our Approach

This section describes our approach in detail. In our method, a given image is first over-segmented into small superpixels, after which segmentation proposals are obtained using two approaches, local and global.

The local approach is basically similar to the greedy superpixel merging technique of [16] except that we use different features. In this approach, each adjacent superpixel pair is assigned a score that represents the visual similarity of the superpixels. The most similar pair is then merged into a single superpixel, and its pixels are saved as a segmentation proposal. Similarity scores of this new superpixel with its neighbors are updated. The merging process is continued iteratively until only one superpixel remains and all the segmentation proposals have been collected. This local approach produces segmentation proposals in all scales efficiently, but is not suitable for discovering objects that consist of dissimilar parts.

For the global approach, we specify the problem as segmentation of foreground from background by minimization of an energy function defined on the superpixel graph. The superpixel graph consists of nodes representing the superpixels and edges representing the neighborhood relations. The energy function has a unary term that is defined by the similarity scores of each superpixel with the foreground and background hypotheses, and a pairwise term that is derived from the similarity scores of adjacent superpixels. By varying the foreground and background hypotheses and parameters of the energy function, we can obtain large numbers of segmentation proposals. Using a training set, we order these sets of parameters so that diverse segmentations are obtained when the number of proposals generated by the global approach is gradually increased.

The following subsections describe the different stages of our method.

3.1. Superpixels and feature extraction

As a first stage of the proposed method, we segment the input image into superpixels using two approaches described in [1] and [8]. The first approach is referred as SLIC and it produces relatively compact superpixels that have approximately equal size. On the other hand, the latter method, referred as FH in the following, produces very diverse set of superpixels that can be anything from half of the image to a narrow object boundary. Both methods have parameters that we set to produce considerable over-segmentation.

Once the superpixels are generated, we compute a feature vector for each of them. In our work, we use SIFT descriptors [13] computed on a dense regular grid and RGB values extracted from each pixel. Both descriptors are quantized using visual vocabulary that is learned using training data. We use 500 and 150 words for SIFT and RGB, respectively. The descriptor for each superpixel is collected by histogramming the visual words from image area it occupies. The corresponding histograms for superpixel i are denoted as h^i_{SIFT} and h^i_{RGB}, respectively.

3.2. Refined superpixels

In the second stage, we pick one of the initial superpixelizations and refine it as follows. We first compute a similarity score for each pair of adjacent superpixels. This score is defined for superpixel pair (i, j) as

$$\begin{aligned}\mathcal{S}(i,j) = & 1 - \frac{1}{\sum_i c_i}\Big(c_1 d(h^i_{SIFT}, h^j_{SIFT}) & (1)\\ & + c_2 d(h^i_{RGB}, h^j_{RGB}) + c_3\mathcal{A}(i,j)\Big), & (2)\end{aligned}$$

where c_1, c_2 and c_3 are constants, $d(a, b)$ is the χ^2 distance between the histograms a and b, and $\mathcal{A}(i, j)$ is the proportion of the image area that superpixels i and j jointly occupy. In our experiments we set $c_1 = c_2 = 1$ and $c_3 = 2$.

Once the scores are computed, the algorithm merges the two most similar superpixels together into a new larger su-

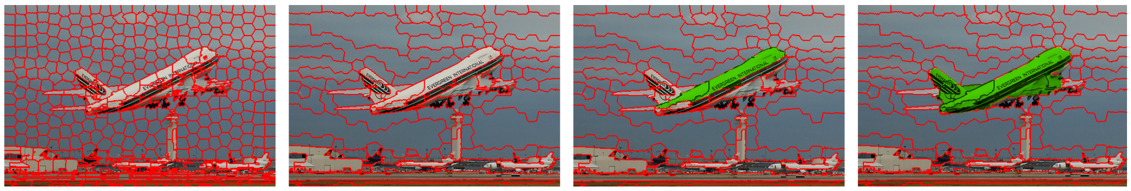
Figure 2. Illustration of different phases in the proposed approach. From left to right: The dense initial superpixelization created using SLIC; the proposed refined superpixelization; and illustration of the superpixel pair merging in the local search; a proposal obtained as a result of one global search branch.

perpixel. After the merge, the scores are updated accordingly. This process is similar to [16], but in our case we do not include the segments generated in the early stages into the output proposal set. Instead, we run the algorithm without storing any of the generated segmentations until a certain threshold for the similarity measure $\mathcal{S}$ is reached.

At the threshold level, we are left with a sparser segmentation that we refer as *refined superpixelization* in the following sections. Figure 2, illustrates one example of the original and refined superpixelizations. The segmentation before the threshold level were discarded, since it was noticed that good object proposals are rarely generated in the early stages. This can be seen by examining the upper bound for the recall which is attainable with the given superpixels after each merging step. The difference in the bound between original and the refined superpixelization was only few percents, but the drop in the number of output proposals was considerable.

3.3. Local search

Once we have reached the similarity threshold, we start collecting the proposal regions by first including all the superpixels in the refined superpixelization in the output set. After this phase the search is split into several parallel branches. One of these branches, continues the merging in the same way as in Section 3.2, but now collecting all the generated segmentations into the proposal set. Figure 2 illustrates one step of the local search algorithm.

This branch is referred as local search, since it considers only superpixel pairs when deciding the next proposal. What often happens, is that this approach fails to detect large non-homogeneous objects that consist of diverse set of superpixels. In such case, there may be some part of the object that is much closer to background appearance and gets merged there before the object is detected. Once this happens, the object is lost from the local search for good. This problem can be alleviated up to some limit by starting from several different initial superpixelizations as in [16]. In our method we used two different superpixelizations as explained in Section 3.1.

3.4. Global search

In order to overcome the limitations of the local search approach, we propose a method that considers all superpixels at once when deciding the next object proposal. This is done by defining an optimisation problem over a graph where each superpixel represents one node and there is a link between each pair of adjacent superpixels. The details are as follows.

For each superpixel, define a label l_i that can take one of the values $\{\mathbf{fg}, \mathbf{bg}\}$, which stand for foreground and background, respectively. Furthermore, define the general form of the energy function as

$$E(L) = \sum_i D(i, l_i) + \alpha \sum_{i,j \in \mathcal{E}} V(i, j, l_i, l_j), \tag{3}$$

where $L = \{l_1, \dots, l_n\}$ is a set of all superpixel labels, $\mathcal{E}$ is the set of adjacent superpixel indices, and D and V are unary and pairwise potentials defined below.

Before going into the actual potentials, we extend the definition of the similarity measure $\mathcal{S}$ to sets of superpixels. The similarity between superpixel sets is simply computed by using (1) with the modification that histograms h^j are computed over the union of the superpixels in the sets. In addition, for global search we drop the size term by setting $c_3 = 0$.

We now define the unary potential D as

$$D(i, l_i) = \begin{cases} \lambda \mathcal{S}(i, C^*_{\mathbf{fg}}) & \text{if } l_i = \mathbf{bg}, i \notin C_{\mathbf{fg}} \\ \mathcal{S}(i, C_{\mathbf{bg}}) & \text{if } l_i = \mathbf{fg}, i \notin C_{\mathbf{bg}} \\ \infty & \text{if } l_i = \mathbf{bg}, i \in C_{\mathbf{fg}} \\ \infty & \text{if } l_i = \mathbf{fg}, i \in C_{\mathbf{bg}}, \end{cases} \tag{4}$$

where $C_{\mathbf{fg}}$ and $C_{\mathbf{bg}}$ are subsets of superpixels that are picked to model foreground and background appearances, respectively. The set $C^*_{\mathbf{fg}}$ is the union of $C_{\mathbf{fg}}$ and its immediate neighbours, as detailed below. In summary, the unary term penalises associating foreground label to superpixels that are similar to background hypothesis and vice versa.

The pairwise potential V is defined as $V(i, j, l_i, l_j) = \mathcal{S}(i, j)$, if $l_i \neq l_j$, and 0 otherwise. This formulation simply penalises the assignments where similar adjacent superpixels are given different labels.

Given the parameters α,λ,$C_{\mathbf{fg}}$ and $C_{\mathbf{bg}}$, we can find the labelling L that minimises the energy function (3). The formulation we use allows to utilise the standard Graph-Cut algorithms to solve the global optimum of (3). This is done very efficiently, since our graph contains only up to few hundred nodes due to the refinement in the superpixelization. Once the optimal labelling is found, the union of the superpixels corresponding to foreground label is included in the set of object proposals.

The formulations above contain four key parameters, α,λ,$C_{\mathbf{fg}}$ and $C_{\mathbf{bg}}$, that allow us to control the generated segmentations. By varying these parameters, we can create thousands of unique segmentation proposals. For the background hypothesis $C_{\mathbf{bg}}$, we use 9 different combinations of the superpixels along the four edges of the image.

Given $C_{\mathbf{bg}}$, we cycle through remaining superpixels, using each of them individually as the foreground hypothesis $C_{\mathbf{fg}}$. Since the histograms of a single superpixel may not capture the characteristics of a large object, we collect the foreground histograms from a wider area. More precisely, we define $C^*_{\mathbf{fg}}$ as the union of $C_{\mathbf{fg}}$ and its immediate neighbours on the superpixel graph, and test the similarity of other superpixels j against it using values of the similarity $\mathcal{S}(i, C^*_{\mathbf{fg}})$.

For each combination of $C_{\mathbf{bg}}$ and $C_{\mathbf{fg}}$, we can additionally vary the parameters α and λ. The former controls the weighting between the unary and the pairwise potentials. We let α vary over the values $\frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \ldots, 3, 4, 5$. The parameter λ is used to control the penalty associated to labeling superpixels similar to foreground hypothesis as background. Large values of λ increase the size of segmentation proposals, which is usually preferable as captures the large objects the are particularly difficult for local search. We uniformly pick several λ values from the interval $[0.75, 3]$.

Using all the parameter combinations, usually over 100 000 unique segmentation proposals per image are generated. However, it was noticed that most of these regions are not complementary and using much smaller subset would result in equal recall values.

Hence, we ordered the parameter combinations $\{C_{\mathbf{bg}}, \alpha, \lambda\}$ by number of objects found on the training set and complementariness. Specifically, we first listed which objects in the training set were found using each triplet. The triplet finding most objects was put on the top of the ordering, and all the objects it found were removed from consideration. Next, we picked the triplet that found most of the remaing objects and removed them. This was repeated until no parameter set found more than 5 previously missed objects. At this point the process was reset by considering all objects again and selecting new sets of parameter triplets (from those not yet selected) by repeating the process. This was done to avoid overfitting to the small training data set. In our experiments each such selection round produced around 14 triplets.

In the experiments, we use 15-150 highest ranking parameter triplets $\{C_{\mathbf{bg}}, \alpha, \lambda\}$. This results in approximately 500-3000 unique segmentations per image. We formed the ordering of the parameter triplets using refined SLIC superpixelizations, but use the same ordering also for global search with FH superpixels.

3.5. Post-processing

Both local and global search algorithms manipulate data purely at the superpixel level with no notion of individual pixels. Because of that, the generated proposals are efficiently identified using superpixel indices. This fact also enables efficient duplicate removal.

4. Experiments

We run the experiments using the 1449 images in the validation set of PASCAL VOC 2012 segmentation challenge[6]. Additionally, we use a subset of 200 images of the training set for learning the parameters for the global search algorithm. The dictionaries for the SIFT and RGB were learned using images outside PASCAL dataset.

To evaluate the quality of the segmentation proposals we use the Pascal overlap vs. union criteria. The overlap score of pixel regions A and B is defined by $m(A \cap B)/m(A \cup B)$, where $m(A)$ is the pixel count of region A. Overlap score gives a real number on the interval $[0, 1]$, value 1 corresponding to perfect segmentation. If the maximum overlap score among a set of segmentation proposals against some ground truth object segmentation exceeds a fixed threshold, we say that the object has been found at that threshold. We also compare bounding boxes of proposals against the bounding box of an object using the same formula, by treating the boxes as rectangular sets of pixels.

Given a threshold, we obtain a recall score for a set of proposals by counting the number of objects for which the highest overlap score exceeds the threshold, and divide it by the total number of objects. We exclude objects annotated as 'difficult' from the test set. We focus on the results for threshold value of 0.7 using the segmentations, but report also some results at threshold level 0.5 as well as using bounding boxes. We plot recall scores as the function of average number of segmentation proposals per image.

4.1. Experiment 1

The first experiment will provide motivation for using the refined superpixelizations as starting point for both global and local search. To obtain the refined superpixelization, we start from a dense superpixelization produced by either one of the algorithms we use [8, 1], and run the greedy pairing process until the highest similarity score among all pairs drops below a certain threshold. Here we show that the coarse refined superpixelization is a better starting point for our global search than the original superpixelization.

In Figure 3, we compare the recall scores of two versions of our algorithm. The first one starts global search

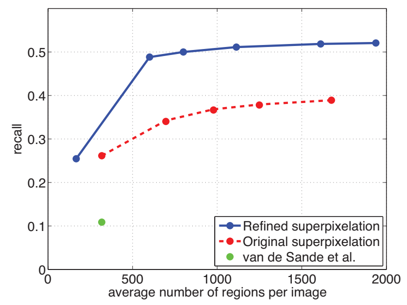

Figure 3. Comparison of recall values using refined and initial FH superpixelizations. Global search works clearly better with refined superpixelizations. Additionally, the figure shows the difference in recall between our similarity measure and the one in [16]

branches at the refined supepixel level whereas the second one starts these directly from the initial superpixels. The starting points of the curves correspond to using the local search only, and subsequent points are obtained by increasing the number of proposals generated by the global search (the parameters for the global search branches are selected according to the prelearnt ordering). Both superpixelizations give similar recall with local search. But it can be seen that for global search, the top curve, which uses refined FH superpixelizations, gives much higher recall than the lower curve, which uses the initial FH superpixelizations.

Here we have used FH superpixels since they were used in [16], but similar results are obtained with SLIC as well. In other words, the results show that the raw superpixelizations from either [8] or [1] are not as good for our purposes as the refined superpixelizations.

4.2. Experiment 2

In this experiment we compare our method against the baselines [5, 4, 16]. We use the publicly available implementations, except for [16]. This is because their executables can not be modified to output segmentations instead of bounding boxes. Hence, we use a re-implementation that was verified to reproduce the results in [16].

Results are shown in Figure 4 where the plots for our method are obtained in similar manner as in Figure 3, but using both FH and SLIC superpixelizations and their combination. The results show that our approach works well with both SLIC and FH superpixels, and combining both of them gives a further improvement in recall. In terms of recall at overlap level 0.7, our algorithm clearly outperforms [16]. We also slightly improve on the methods of [5, 4], which are tens of times slower than our method.

We also ran our local search using the settings comparable to those presented in [16], i.e. utilising up to 4 color spaces and two FH superpixelizations in each color space (not refined). The leftmost points of the cyan and green curves correspond to using just the RGB color space and one superpixelization, while the rightmost points collect local segmentation proposals by running the respective algorithm 8 times utilizing 4 different color spaces and 2 different FH superpixelizations. The features of both methods change when the image is transformed into another color space, resulting in different superpixel mergings and therefore different proposals.

Method	this paper	van de Sande et al. [16]
Oxford Sculptures	0.91 / 0.79 / 1198	0.87 / 0.58 / 1913
Oxford Flowers	0.96 / 0.95 / 1879	0.94 / 0.71 / 2200
UCSD Birds	0.94 / 0.43 / 1204	0.82 / 0.23 / 1862
CORE	0.87 / 0.60 / 890	0.86 / 0.40 / 2071

Table 1. Recall values at 0.5 and 0.7 overlap thresholds and the average number of regions per image for Oxford Sculptures & Flowers, UCSD birds, and CORE.

The leftmost points of the cyan and green curves show that our features are substantially better in the case of using just single RGB version, but the rightmost indicate that our local search does not benefit as much from addition of the color spaces as [16] does. This may be because we did not focus on making our features applicable outside RGB. In any case, in our situation it is more profitable to vary the initial superpixelizations, rather than the features in order to diversify the search. The usage of single set of features is also beneficial to keep the global search efficient.

4.3. Experiment with other datasets

In order to demonstrate that the proposed method generalises beyond PASCAL VOC, we carried out experiments using four additional datasets: Oxford sculptures[3] & flowers[4], UCSD birds[5], and CORE[6]. In all cases, the parameters were the same as for VOC and the overlap measure for the evaluation was computed using ground truth segmentations. Table 1 shows the results for our method using only SLIC superpixels and the method of [16]. In all the four cases the average number of proposal regions per image was notably smaller with our method than with [16] although the recall of our method was clearly higher. The results are interesting also because the object classes in these datasets are quite different to those in VOC.

4.4. Comparison of execution times

Table 2 compares the execution times of the proposed approach and the baselines. The results indicate that the methods of [5, 4] are very slow, but achieve recalls close to our method. The approach of [16] essentially runs the local search, similar to ours, 8 times, using computationally inexpensive features. The implementation we used for [16] uses Matlab MEX functions for the computational bottleneck (our method does not), which probably explains the

[3] http://www.robots.ox.ac.uk/~vgg/data/sculptures6k/
[4] http://www.robots.ox.ac.uk/~vgg/data/flowers/102/index.html
[5] http://www.vision.caltech.edu/visipedia/CUB-200.html
[6] http://vision.cs.uiuc.edu/CORE/

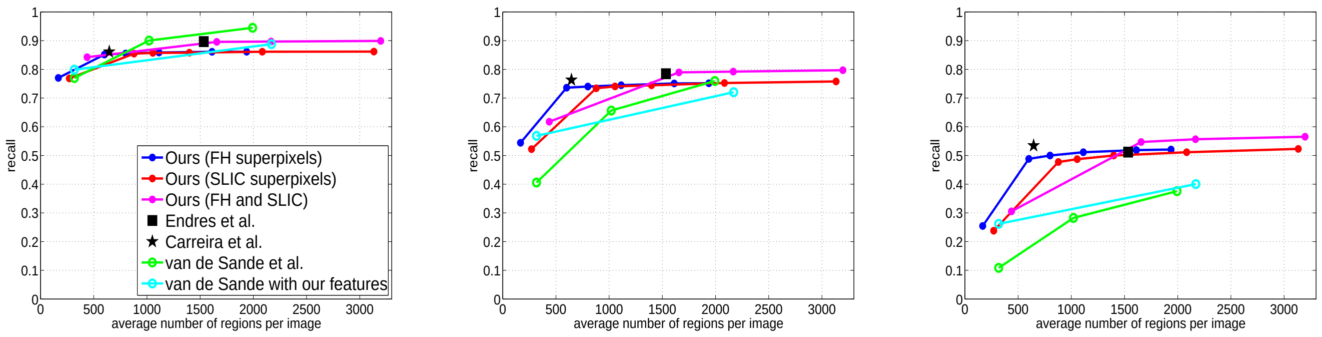

Figure 4. The recall values for various methods using bounding boxes with overlap threshold 0.5 (left) and segmentations with overlap thresholds 0.5 (middle) and 0.7 (right), respectively. Our method using two superpixelizations improves over the slower methods of Endres and Carreira, and substantially improves the accuracy of 0.7 segmentations in comparison to van de Sande. Using one superpixelization per image is faster and gives reasonable recall as well.

Method	Recall	Regions	Time
This paper (FH and SLIC)	0.546	1656	13.5
This paper (FH)	0.488	601	6.5
Carreira10 et al. [4]	0.534	646	280
Endres et al. [5]	0.512	1534	140
van de Sande et al. [16]	-	-	7.5
reimplementation of [16]	0.376	1993	2.5

Table 2. Recall values for VOC 2012 segmentation challenge validation data using 0.7 overlap threshold, with the average number of proposals and average running time (seconds) per image. For our method, we use 15 parameter triplets and initial superpixels as indicated in the brackets. These correspond to the second points from left of the magenta and blue curves in Fig. 4.

speed difference to the original implementation.

The breakdown of computation time for our method in the 6.5 seconds case, which uses a single FH superpixelization and does both local and global search, is as follows. About 57% of time goes to computing the SIFT descriptors, for which we use VLFeat[7] functions. Decreasing the histogram size from 500, making the computation grid on image sparser, or using altogether different features would be options to speed up the method. However, we observed that the combination of SIFT with the color feature produced considerably better results than either feature alone. Even if multiple superpixelizations are used, these features only need to be calculated once per image, while using multiple color spaces, as [16] does, might require different features for each case. Furthermore, dense SIFT features are often needed in the subsequent detection phases and are computed in any case.

The aforementioned bottleneck, computation of histogram distances to obtain the similarity scores, takes about 19% of the execution time. We used the χ^2 distance, which gave good results in comparison to other faster options. Finally, about 15% of the time goes to constructing the initial superpixelizations, depending on the method used.

[7] http://www.vlfeat.org/

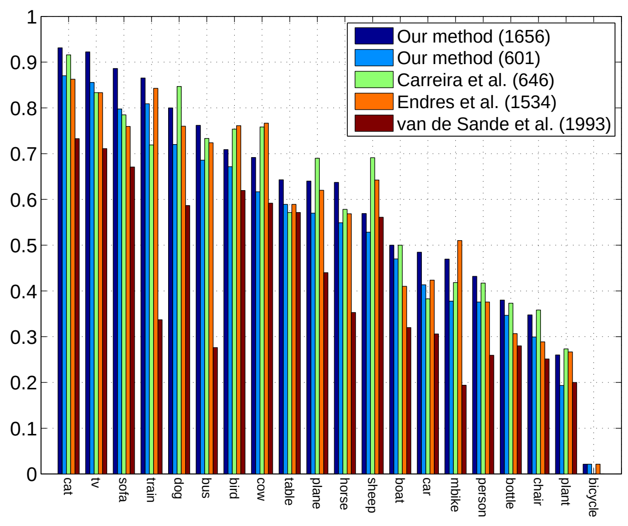

Figure 5. Recall values for each object class in VOC 2012 validation data using segmentations and 0.7 overlap threshold. Numbers in brackets are average numbers of proposals used.

5. Conclusion and Future Work

In this paper, we have presented a fast approach for generating high-quality class-independent object segmentation proposals for color images. One of the key innovations of the proposed method is to combine techniques for global and local grouping of superpixels in such a manner that high recall of objects can be obtained with a relatively low number of regions, even at high values of the Pascal overlap threshold. Our experimental evaluation with annotated Pascal VOC images shows that the generated region proposals provide accurate segmentations for various kinds of objects. Our approach is approximately as fast as the fastest available comparison method but provides substantially more accurate segmentations. In terms of segmentation accuracy our approach provides comparable results with the best comparison methods but is more than 10 times faster.

Considering future work, we believe that our method has

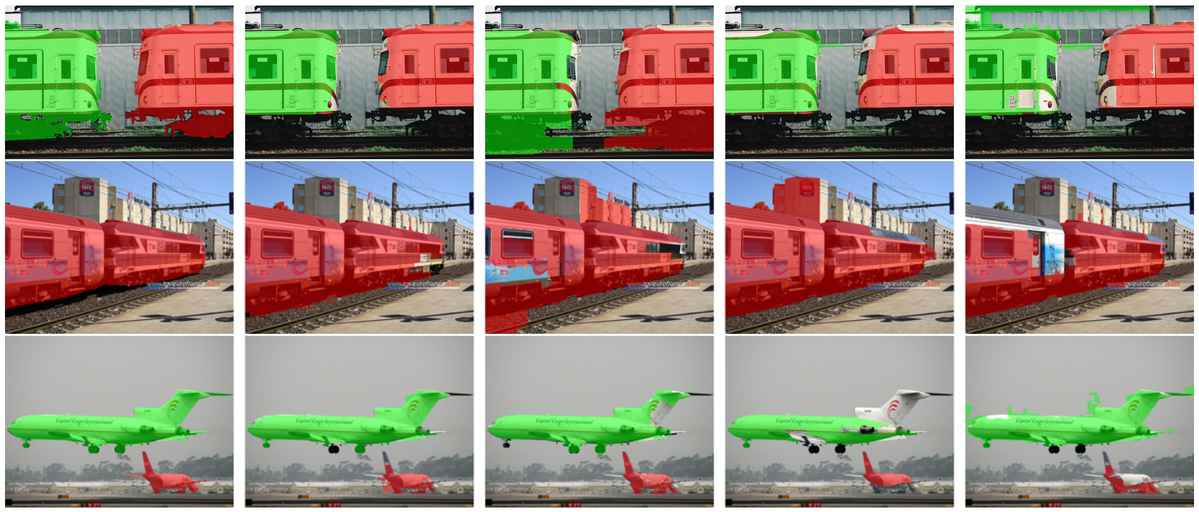

Figure 6. Examples of best detections for each method (i.e. the region with the highest overlap is shown for each ground truth object). From left to right: ground truth, our method (on average 1656 regions per image), Carreira et al. (646 regions), Endres et al. (1534 regions), van de Sande et al. (1993 regions). More examples can be found from the supplementary material.

a lot of potential to be used as a component in object detection and segmentation systems, such as those recently submitted to ImageNet and Pascal VOC challenges. This belief is based on the fact that until now [16] has been the only approach which is able to produce accurate segmentation proposals sufficiently fast in order to be applicable for very large datasets, such as the ImageNet. Yet, the advantages of this kind of an approach have been clearly demonstrated as the contributions from the authors of [16] have won the detection challenges in ImageNet 2011 and Pascal VOC 2012. Since our approach provides more accurate segmentations than [16], it would be interesting see whether it can further improve the performance of detection systems as well. In addition, since our method is substantially faster than [4] which is successfully used in the state-of-the-art segmentation systems [9, 3], addressing segmentation of large image sets might be one fruitful research topic in future [10]. In order to advance the future use of our approach we will make our implementation publicly available upon the publication of the article.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 2012. 2, 3, 5, 6

[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, 2010. 1

[3] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*, 2012. 8

[4] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *CVPR*, 2010. 2, 3, 6, 7, 8

[5] I. Endres and D. Hoiem. Category independent object proposals. In *ECCV*, 2010. 2, 3, 6, 7

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *IJCV*, 2010. 1, 5

[7] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010. 1, 2

[8] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004. 2, 3, 5, 6

[9] A. Ion, J. Carreira, and C. Sminchisescu. Image segmentation by figure-ground composition into maximal cliques. In *ICCV*, 2011. 8

[10] D. Kuettel, M. Guillaumin, and V. Ferrari. Segmentation propagation in ImageNet. In *ECCV*, 2012. 8

[11] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *TPAMI*. 2

[12] F. Li, J. Carreira, and C. Sminchisescu. Object recognition as ranking holistic figure-ground hypotheses. In *CVPR*, 2010. 3

[13] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 3

[14] E. Rahtu, J. Kannala, and M. Blaschko. Learning a category independent object detection cascade. In *ICCV*, 2011. 1, 3

[15] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 2004. 2

[16] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as selective search for object recognition. In *ICCV*, 2011. 1, 2, 3, 4, 6, 7, 8

[17] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009. 1, 2

[18] Z. Zhang, J. Warrell, and P. H. S. Torr. Proposal generation for object detection using cascaded ranking SVMs. In *CVPR*, 2011. 3