A Fast and Simple Algorithm for Producing Candidate Regions

Boyan Bonev and Alan Yuille

University of California, Los Angeles
bonev@ucla.edu, yuille@stat.ucla.edu

Abstract. This paper addresses the task of producing candidate regions for detecting objects (e.g., car, cat) and background regions (e.g., sky, water). We describe a simple and rapid algorithm which generates a set of candidate regions $\mathcal{C}_\mathcal{R}$ by combining up to three "selected-segments". These are obtained by a hierarchical merging algorithm which seeks to identify segments corresponding to roughly homogeneous regions, followed by a selection stage which removes most of the segments, yielding a small subset of selected-segments $\mathcal{S}$. The hierarchical merging makes a novel use of the PageRank algorithm. The selection stage also uses a new criterion based on entropy gain with non-parametric estimation of the segments' entropy. We evaluate on a new labeling of the Pascal VOC 2010 set where all pixels are labeled with one of 57 class labels. We show that most of the 57 objects and background regions can be largely covered by three of the selected-segments. We present a detailed per-object comparison on the task of proposing candidate regions with several state-of-the-art methods. Our performance is comparable to the best performing method in terms of coverage but is simpler and faster, and needs to output half the number of candidate regions, which is critical for a subsequent stage (e.g, classification).

Keywords: Hierarchical grouping, segments selection, candidate regions

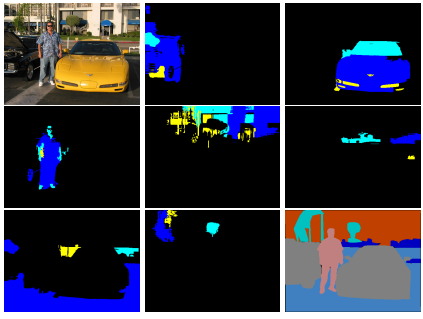
1 Introduction

The goal of this paper is to propose a fast and simple way to generate a set $\mathcal{C}_\mathcal{R}$ of *candidate regions* which can correspond to objects and background regions (e.g., sky and water) in images, in contrast to methods which only focus on foreground objects [2]. The output of our method is illustrated in figure (1a) which shows candidate regions for car, person, building, and background classes, like ground, grass, building. The candidate regions are obtained combining one to three of a set of *selected-segments*, shown in different colors. Observe that the candidate region for the person contains segments for the head, torso and legs. By contrast, the candidate region for the car contains segments for the car body, the lights, and the window.

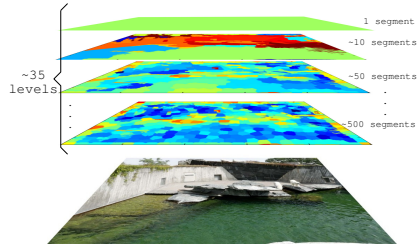
There has been recent work on detecting candidate regions for objects. The Ultrametric Contour Map (UCM) algorithm has been extended to perform this task [4]. Other methods include the highly successful Constrained Parametric Min-Cuts (CPMC) [7], Selective Search [22], and the Segmentation by Weighted Aggregation (SWA) algorithm [12] and its variants [3, 23] (the SWA family of algorithms output salient segments

which can be thought of as candidate regions). The detection of candidate regions can be followed by a classification stage, for example see [4, 7].

In this paper, we specify a fast and simple method for obtaining candidate regions $\mathcal{C}_{\mathcal{R}}$ for *both objects and background regions*. We start by constructing a hierarchy of segments $\mathcal{S}_{\mathcal{H}}$, see figure (1b), using a hierarchical merging algorithm. A novel aspect of this grouping algorithm is the use of a PageRank [11] based criterion to select the order of segments to merge. This enables the algorithm to rapidly merge background segments (e.g., the sky) into large segments while maintaining small segments corresponding to birds or airplanes. This segment merging process results in segments which have roughly homogeneous image properties, except at the highest levels where objects and background are merged. Next, we use a selection criterion which aims to remove segments which are subparts of larger homogeneous regions and yield a subset of "selected-segments" $\mathcal{S}$. To do this we use a novel entropy gain criterion which involves introducing a non-parametric method for estimating the entropy of image segments. Our method is fast and runs in 4s per image in a Matlab implementation.



(a) Examples of candidate regions for objects and background regions. Left-to-right and top-to-bottom: image, top three selected-segments for left car, right car, person, building, grass, ground, trees, and ground truth. Most objects are covered well by two to three selected-segments.



(b) Multiple levels in a hierarchy. Segments with a good coverage of objects or parts may happen at different levels. 80% to 90% of the segments can be discarded because they go across boundaries of objects or because they don't cover a large area of an object.

Fig. 1: (a) Examples of candidate regions composed by three selected-segments; (b) Hierarchy of segments.

To evaluate our method we study its performance on a dataset with 57 labels for objects and background regions. This is the Pascal VOC 2010 dataset where we have assigned every pixel one of 57 labels. We demonstrate that three segments are typically sufficient to provide good coverage of most objects and background regions. The coverage given by the subset of selected-segments is only 0.2% lower. Next we test the effectiveness of small combinations of our selected-segments (one to three segments) to cover objects and background regions (where coverage is measured by intersection over union). We report good results which are roughly similar to the evaluation of the

candidate regions provided by [4]. Our method has the advantage of being simple, fast, and based on two novel ingredients.

This paper is organized as follows. In section 2 we discuss our use of PageRank and non-parametric entropy, which are the most novel aspects of our method. Section 3 describes how we generate the hierarchy of segments and how we select a smaller set of selected-segments from the hierarchy. Our next contribution is in section 4, where we present an analysis of why we need combinations of up to three selected-segments (subsection 4.2). Based on that we generate candidate regions in subsection 4.3 and we test on 57 classes our candidate regions and present a comparison with [4].

1.1 Related work

Our work has been inspired by previous work on detecting candidate regions. This includes a variety of methods which first construct a hierarchy of segments and then use a procedure to select important, or in our words "selected"-segments. An early attempt [19] which outputs $10,000$ candidates per image, analyzes the importance of spatial support for the segments. A successful recent example is based on the UCM hierarchical algorithm [5, 4], which is based on gPb [18]. This method is particularly relevant because it outputs candidate regions for objects and background regions. It uses a combination of three segments output by the UCM algorithm but they do not describe their motivation for choosing this number nor do they give a detailed description of how they select segments and obtain their candidate regions (but they kindly supplied us with these regions for our experiments).

We perform an analysis on the number of segments which cover a foreground or background class. We have not found a similar attempt in the literature. However, our analysis is consistent with [5, 4] where the authors make 3-combinations to improve the coverage of their candidate regions. In that work they do not elaborate on the reason for choosing $k = 3$ and they do not specify how they select their segments. Here we experimentally show that 3 is adequate for both foreground and background classes. In our comparison to the candidate regions of [4], the coverage we obtain is slightly lower by using nearly half the number of candidates. This simplifies the task of any further high-level task on top of our candidate regions $\mathcal{C}_{\mathcal{R}}$. Also, our method is two orders of magnitude faster. This is mainly because their method is based on the output of gPb. In this paper, we do not address the task of classifying the segments – unlike [5, 4].

Related hierarchical grouping methods include the SWA algorithm [12] and a recent variant [3]. This method has also been extended to video segmentation [23]. After the grouping, it uses a local saliency measure, based on large affinity within the segment and small affinity across its boundaries, to detect salient segments. These methods however, tend not to do a further step of grouping the segments into combinations (as done in [5, 4]) and they have been mostly evaluated for finding segments which cover foreground objects, see [3]. A related method which has been also evaluated on covering objects is [22]. There is also related work by [10]. We should also mention hierarchical segmentation which has been used to learn models of objects [21]. But although these approaches have some commonalities with ours they differ in goals and methods (like PageRank or the entropy-based criterion).

Selective Search [22] is a fast method (4s as our method) which gives a very high recall, 98%, on bounding boxes (2,100 candidate boxes). However it is not optimized for proposing candidate masks. It groups an initial set of segments given by [10]. They group them based on appearance properties (color, texture), the size and compactness of the segments.

The CPMC method outputs a set of foreground segments [7]. This method is very successful but restricts itself to objects and does not attempt to find segments for background regions. It proceeds in several stages. The first stage uses repeated applications of a max-flow algorithm to output a set of foregrounds segments based on groupings of an edge map provided by gPb. This set of segments is pruned by non-maximal suppression. The second stage ranks these segments using cues trained on the Pascal VOC dataset. The first stage of this algorithm is most similar to ours (i.e. it uses only low level cues). Their method is one to two orders of magnitude slower than ours, which makes it inapplicable to large scale datasets like ImageNet [9].

A key element of our approach is the use of an entropy gain criterion for selecting segments. This includes using a non-parametric multivariate entropy estimator which is due to [13, 20, 14]. This estimator has been applied to estimate the entropy of natural images by Chandler and Field [8] but is not widely used in vision. A different non-parametric entropy estimator has been used for model-selection in clustering, including an example of color segmentation [6]. In [16] entropy rate is used for segmentation but the purpose is different: they quantify the stochastic process over the graph, favoring formation of compact and homogeneous clusters. [17] describes a method for segmenting data by encoding criteria. The idea of good encoding relates to our entropy criterion for selecting segments. But our goal is to detect maximal homogeneous segments which are used for later processing and not to segment data.

2 PageRank and Selection by Entropy Gain Criterion

The method in this paper first constructs a hierarchy of segments and then selects a subset of selected-segments. From these we construct candidate regions $\mathcal{C}_{\mathcal{R}}$ which are combinations of up to three selected-segments $\in \mathcal{S}$. Our method contains two novel and simple ingredients. The first is the use of the PageRank algorithm to create the hierarchy $\mathcal{S}_{\mathcal{H}}$ by grouping segments. The second is the selection of segments based on an entropy gain criterion using a non-parametric estimate of the entropy of the segments.

2.1 Grouping segments by the PageRank algorithm

Our approach groups only a fraction of the segments at each hierarchy level $\mathcal{S}_l$. In that way it allows large homogeneous areas to grow "faster", coexisting with small segments. The PageRank algorithm [11] is used to determine which of the segments are grouped at each level. Its use is motivated by the desire to start by merging those segments which have similar statistics to their neighbors. PageRank has a straightforward application in our framework because we define an asymmetric dissimilarity measure between the segments, see section (3.1) for details.

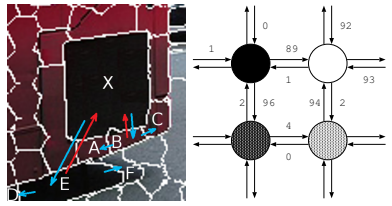
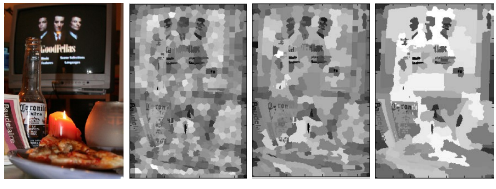


Fig. 2: Left: Asymmetric dissimilarity function. For X, the most similar are B and E (2^{nd} neighbor) because they don't modify a lot the statistics of X. However, X is not the most similar for B and E, from their perspective other merges are preferable. Right: The PageRank algorithm prioritizes merging segments which are very similar to their neighbors. In this example, the white node (representing a segment) has highest ranking and so is selected first for merging. The edge weights denote (asymmetric) similarity between the segments.

PageRank is illustrated in figure (2)-right. It plays a key role in ensuring that our segments take *global properties* of the image into account when merging segments to construct the hierarchy. At each level l of the hierarchy, it selects which segments should be allowed to merge, by taking into account the whole *directed* graph, where the nodes are the segments in $\mathcal{S}_l$ and the weights of the directed edges are given by the asymmetric dissimilarity, which we define in (3). In particular, this strategy also allows some segments to grow "faster", that is, the segments $\mathcal{S}_l$ of a level l may have different sizes. In that way l is not directly related to the size of the segments (as in [12]), as the appearance also determines the size of the segments. In figure (3a)-right it can be seen that small salient areas can "survive" together with large segments covering homogeneous areas. This is an advantage when searching for region candidates for objects of different sizes and with large background regions.



(a) PageRank output: at each level we merge first the highest rank (whitest) nodes. Note that the "salient" regions have lower rank, while the segments covering homogeneous areas tend to have higher ranks.

(b) The effect of PageRank. Top: original image. Left: best segment without PageRank. Right: best segment using PageRank.

Fig. 3: (a) PageRank output at different levels of the hierarchy; (b) Example of the effect of PageRank.

More formally, PageRank quantifies the importance of each segment (i.e. graph node) after a sequence of probabilistic transitions over the graph. These probabilistic

transitions are encoded by a stochastic matrix. At level l, W_l is a stochastic matrix of size N_l where each element is given by the similarity from node i to j, normalized by rows so that the outgoing edges form a probability distribution, $\forall i,\ \sum_j^{N_l} w_{ij} = 1,\ i,j \in [1, N_l]$. Given that we work with the dissimilarity $\Delta_{i|j}$ further defined in (3), we need to invert it to represent a similarity value. We define

$$w_{ij} = \frac{\Delta_{i|j}^{-1}}{\sum_{j=1}^{N_l} \Delta_{i|j}^{-1}}.$$

PageRank returns a ranking of the nodes and we select those nodes with the highest ranks for merging. The merging order does not determine to which of its neighbors a node is merged: for this we use $\Delta_{i|j}$ as explained in section 3.1.

2.2 Non-parametric entropy and the entropy gain criterion

After constructing the segment hierarchy $\mathcal{S}_\mathcal{H}$we need a criterion to obtain the set of selected-segments $\mathcal{S}$. This criterion should ideally select segments which are roughly homogeneous and are as large as possible. We use the entropy of each segment as a measure of its homogeneity. Now suppose that two segments are combined in the hierarchy to form a larger "parent" segment. We compute the difference between the entropy of the parent segment and the entropy of its two children. If the difference is small, then we do not select any of the two segments because we consider them to be subparts of a larger homogeneous region. If the difference is big, then we select the two child segments because we cannot grow them further. This is illustrated in figure (4)-top where segments A and B can be merged (because the entropy gain is small) but where segment C cannot be merged with A or B because the entropy gain would be too big.

We emphasize that the selected-segments can overlap. This is illustrated in figure (4)-bottom where the segments at small scales correspond to windows and are selected individually. The wall of the house (without the windows) is also a selected-segment. Finally the whole house is another selected-segment because at this scale the windows form a semi-regular texture pattern.

This criterion is intuitive but requires a method for computing the entropy of a segment. This is challenging because we do not have a parametric probability distribution for the segment variables from which to calculate the entropy (our data is poorly fit by standard distributions like Gaussians) and, even if we did, for small segments we may not have enough measurements to estimate the distribution. Instead we use a non-parametric method [13, 20, 14] which has been previously used to measure the entropy of natural images [8]. The power of this method is the ability to estimate entropy in a multivariate way without assuming any kind of independence among the different dimensions of the vector of statistics. It is applicable to small samples because it bypasses the density estimation, working directly on distances between samples.

The estimator of the entropy from n samples $\{x_1, ..., x_n\}$ in p-dimensional space is given by:

$$H_k^n(f) = \frac{p}{n} \sum_{i=1}^{n} \log R_{i,k,n} + \log \frac{\pi^{p/2}}{\Gamma(\frac{p}{2}+1)} - \frac{\Gamma'(k)}{\Gamma(k)} + \log n, \qquad (1)$$

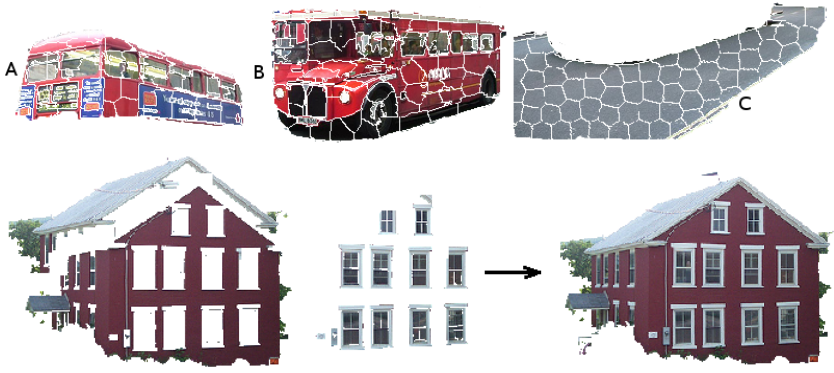


Fig. 4: Top: Entropy gain: When segments A and B are merged, the increase of entropy is not as big as if they were merged with C. The entropy is calculated from the small (first-level) segments which are shown in the figure. Bottom: Selected-segments at different scales. The walls of the house (left panel). Each of the windows (center panel). The whole house (right panel).

where $R_{i,k,n}$ is the Euclidean distance between x_i and its k^{th} nearest neighbor in the set $\{x_1, ..., x_n\}$, and $\Gamma(.)$ is the Gamma function.

This non-parametric estimator is asymptotically unbiased and consistent [20, 14]. It can be derived in three steps. Suppose the unknown probability density function is $f(x)$. The first step estimates its entropy $-\int f(x) \log f(x) dx$ by the normalized sum $\frac{1}{N} \sum_{i=1}^{n} \log f(x_i)$ of the logarithm of the probability density $f(.)$ evaluated at a random sample $x_1, ..., x_n$ of points. The second step estimates $f(x_i)$ by a k- nearest neighbor method by

$$f(x_i) \approx k/n\Gamma(p/2+1)\frac{1}{\pi^{p/2} R_{i,k,n}^{p}},$$

where $R_{i,k,n}^{p}$ is the Euclidean distance between x_i and its k^{th} nearest neighbor in the set $\{x_1, ..., x_n\}$. The third step makes an additive correction to this estimator to ensure that it is statistically unbiased, see Theorems 8,9 in [20]. The key intuition behind this method is that the nearest neighbors of random samples can be used to give a local estimate of the probability density (step 2).

The estimator can be applied to image segments by setting the samples x_i to be the appearance features (e.g., color and texture) calculated in the lowest-level segments within the segment (i.e. those at the bottom of the segment hierarchy). See section (3.2) for more details. This method has been only used once in vision, to our knowledge, by Chandler and Field [8] to estimate the entropy of natural images.

3 Method

Our method has two stages. The first constructs a hierarchy of segments, as described in section (3.1). The second uses an information gain criterion to obtain a subset of

selected-segments. The characteristics of both stages is that they are simple and fast. The ingredients are simple image features, an asymmetric similarity measure including an edge term, the PageRank algorithm, and the entropy gain criterion for selection.

3.1 Constructing the Segmentation Hierarchy

The algorithm starts with a basic set of small segments $\mathcal{S}_1$ which are produced by an over-segmentation method. In this work we use the SLIC algorithm described in [1] because of its simplicity, its speed, and because it allows segments with irregular shape. Then we build a segmentation hierarchy $\mathcal{S}_\mathcal{H} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_L$ of L levels by merging segments as follows.

Each segment is described by an *appearance vector*

$$V = (\overline{\mu}, \overline{\sigma}, c_x, c_y, w, h), \tag{2}$$

where $\overline{\mu}, \overline{\sigma}$ are the mean and the standard deviation of the Lab color space components and the first and second derivatives on the l channel, $(l, a, b, \nabla_x, \nabla_y, \nabla_x^2, \nabla_y^2)$. Here (c_x, c_y, w, h) are the centroid of the segment and the dimensions of its bounding box. These appearance vectors are designed so that they can be computed efficiently recursively for new segments composed by merging.

We define an asymmetric dissimilarity function $\Delta^A_{i|j}$ between segments at the same level which are 1^{st} or 2^{nd}-order neighbors. The appearance term, see figure (2)-left, is defined to be

$$\Delta^A_{i|j} = ||V_i - V_{i \bigcup j}||_2.$$

This is the change in the appearance vector of region i caused by merging it with region j. This function is asymmetric – i.e. $\Delta^A_{i|j} \neq \Delta^A_{j|i}$. This terms will encourage merging neighboring regions which have similar appearance vectors. The asymmetry has the meaning that after merging i and j, the statistics of i may be modified in a different degree than the statistics of j. Intuitively, each segment has its own preference for merging or not to a neighbor, and it is based on minimizing the change of appearance.

The appearance dissimilarity function is modified by an edge-term ($E_{i,j} \in [0, 1]$) that represents the amount of edge-ness on the boundary between two adjacent regions. This edge term is computed only once. Any fast edge detector can be used. We obtained good results using the Sobel detector and obtained slight improvements (1.7%) when we switched to the Sketch-Token method [15] (used for the experiments here). The intuition for this edge modulation is that we penalize the similarity between adjacent regions if there is an edge between them. We do not introduce an edge-term between segments in the 2^{nd}-order neighborhood (because we want to allow this type of merging to jump between regions) and instead we pay a fixed penalty of size 1 (which is the maximum value the edge term can take).

This gives an asymmetric dissimilarity function $\Delta_{i|j}$:

$$\begin{aligned} \Delta_{i|j} &= E_{i,j} + \Delta^A_{i|j}, \text{ if } i, j \text{ are } 1-\text{neighbors}, \\ \Delta_{i|j} &= 1 + \Delta^A_{i|j}, \quad \text{if } i, j \text{ are } 2-\text{neighbors}. \end{aligned} \tag{3}$$

Then we use the PageRank algorithm [11] to rank the need of pairing between segments (i, j) based on this dissimilarity function. We allow the top-ranked 30% segments to merge, unless they violate the condition $\Delta_{i|j} > 0.9\Delta_{j|i}$. The intuition is that this ranking encourages merging between segments which are most similar, but that we reject merges in situations where the dissimilarity function between two regions is too asymmetric. After these merges, on the next level of the hierarchy, we re-compute the PageRank algorithm and repeat the process. See ranking examples in figure (3a). Finally, the last level $\mathcal{S}_L$ contains a single segment with the whole image.

Our algorithm does not have an explicit mechanism for handling textures. However it usually merges textures into a single segment. In the case of textures which are finer than the size the first level $\mathcal{S}_1$ segments, the texture characteristics are captured by the mean and standard deviation of the segments and they are likely to be merged. In the case of very coarse textures several big and disconnected segments may be formed.

3.2 Obtaining Selected-segments

The next stage of our method requires obtaining a set $\mathcal{S}$ of selected segments which are roughly homogeneous but are as large as possible. This is particularly challenging because the sizes of objects and background stuff varies considerably and so we need to select segments at different scales. Also in some cases, see figure (4)-bottom, we need selected-segments that are overlapping and represent structure at different scale (e.g., the windows at one scale, the whole building at another).

The total number of segments $|\mathcal{S}_\mathcal{H}|$ produced by our algorithm largely depends on the number of segments of the initial over-segmentation, $|\mathcal{S}_1|$. In this work we set $|\mathcal{S}_1| \approx 600$ which results in $L \approx 35$ levels in the hierarchy with an average total of $|\mathcal{S}_\mathcal{H}| \approx 1200$ segments. We estimate that only 5% to 10% of the segments are really needed with which to form candidate regions (by small groups). We propose an entropy gain criterion to select this set $\mathcal{S} \subseteq \mathcal{S}_\mathcal{H}$ of segments. The criterion is required to obtain good candidate regions for both object and background regions.

More precisely, we establish a threshold D for the entropy increase d after merging two segments $s_i, s_j \in \mathcal{S}_l$ with area sizes $a_i,\ a_j$ into a new one $s_m = s_i \cup s_j \in \mathcal{S}_{l+1}$ at the next level $l + 1$. We define the entropy gain as:

$$d = H(s_m) - \{H(s_i) + H(s_j)\}\,. \tag{4}$$

Here $H(s)$ is the multivariate entropy of the appearance vectors V' of the first level segments $s_f \in \mathcal{S}_1$ which compose the segment s, that is, $\bigcup s_f = s$. Similarly to (2), the appearance vector is $V = (\overline{\mu}, \overline{\sigma})$, where $\overline{\mu}, \overline{\sigma}$ are the mean and the standard deviation of $(l, a, b, \nabla_x, \nabla_y, \nabla_x^2, \nabla_y^2)$. The non-parametric estimation of $H(\cdot)$ was described in section (2.2).

The entropy gain d is not always positive, but we have experimentally observed that it usually is, in natural images. In very homogeneous areas or repeating patterns, d can be negative.

The selection criterion is as follows. For smooth increases $d < D$ we do not select the segments. For increases $d \geq D$ we select the two segments because at the next level they are merged into a more heterogeneous segment. See an example of a selected-segment in figure (5)-third image. Note that we use the entropy value as a threshold for

checking whether a segment maintains a certain amount of homogeneity. But it is not used for deciding to which segment to merge, as for this purpose we use the appearance vectors of the segments, which provide richer information.

Fig. 5: Left-to-right: Original image; a segment that is too small for the region it could cover; a selected-segment that covers a large homogeneous area, a segment that covers a still larger but heterogeneous area.

We fix a threshold D such that the amount of selected-segments is approximately $|\mathcal{S}| \approx 0.1|\mathcal{S}_{\mathcal{H}}|$. We reduce the set of segments from 1200 to 116 on average. This causes a minimal loss: 0.2% less of IoU with respect to not applying any selection criterion. The loss occurs more often on small objects than on large ones.

4 Experiments

4.1 The Dataset

In our experiments we perform analysis on the PASCAL VOC 2010 detection set. There are 57 classes among which 15 are background and the rest are foreground objects. See the list of class labels in figure (6). The 57-class dataset was constructed by hand-labeling the images so that every pixel is assigned to one of 57 object classes or to a default class (this ground truth is being prepared for publication and will be made available if this paper is accepted). The labeling was done by supervised students (i.e, not Mechanical turk).

The objects are placed in a rough taxonomy of four classes.

- Natural objects: bird, cat, cow, dog, flower, horse, mouse, person, sheep.
- Large homogeneous regions: grass, ground, mountain, rock, sky, water, snow, tree, wood, wall.
- Man-made (indoor): bag, clothes, tv monitor, computer, keyboard, bed, book, bottle, cup, plate, food, cabinet, shelves, chair, sofa, table, dining table, ceiling, door, floor, light, potted plant.
- Man-made (outdoor): aeroplane, train, truck, bicycle, motorbike, boat, car, bus, bench, building, fence, road, sidewalk, track, sign, window.

There is also a ”default background” class for pixels which cannot be assigned to any of these 57 categories. The number of pixels assigned to this class is negligible and we ignore this class for the results presented here.

4.2 Best segments analysis

We first investigate how many of our segments are needed to cover an object or background region, where coverage is the region-based intersection over union (IoU). Most objects consist of several roughly homogeneous regions (e.g., the face, the shirt, and the trousers of a man can each be roughly homogeneous but the complete man is not). Hence, to get good coverage of an object we need to combine several segments.

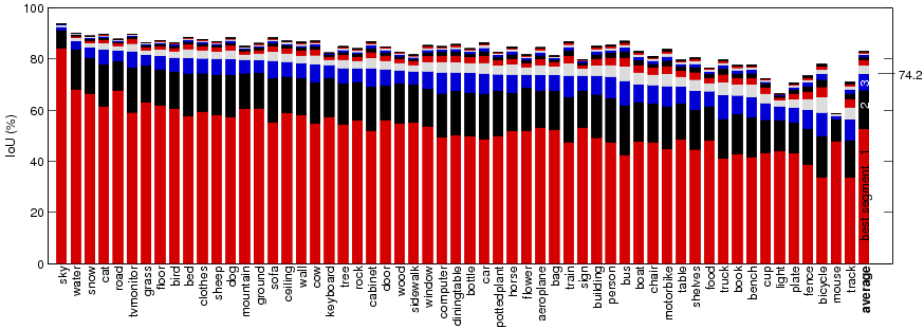


Fig. 6: Overall coverage of the best segments for each object class. Red, black, and blue shows the average proportion of the object covered by the first, second, and third best segments respectively. Observe that the amount of coverage by the first segment ranges from 35 percent (for track and bike" to almost 90 percent for sky. The proportion covered by the first three segments is typically between 70 and 75 percent with a few outliers (e.g., track and sky).

The coverage by our segments is illustrated in figure (6). This displays the proportion of the object that is covered (IoU) as a function of the number of segments (from one to ten). In this experiment we do not propose selected-segments but we test them all with the ground truth to quantify the optimal performance that could be achieved for the provided set of segments $\mathcal{S}_{\mathcal{H}}$. The segments used for this analysis do not overlap, even if belonging to different level $\mathcal{S}_l$. To evaluate if a segment is covering a part of an object we require that at least 80% of it has to be within the ground-truth shape of the object. We use the combination of segments which give optimal coverage for the given segmentation hierarchies.

The results show that our segments have good performance for many object classes in the sense that three segments give over eighty percent of coverage (on average). In particular, good performance is obtained for: (I) All natural objects except for mouse and person (bird, cat, cow, dog, horse, sheep, flower). (II) All large homogeneous regions (grass, ground, mountain, rock, sky, water, snow, tree, wood, wall). (III) A subset of the man-made indoor objects – bag, bed, ceiling, computer, door, sofa, TV monitor, (IV) A subset of the man-made outdoor – airplane, sign, window. The good performance categories have fairly homogeneous appearances and they are typically of medium to large size in the images (e.g., the mouse has worse performance mainly because it tends to appear in small sizes).

Performance is medium quality on object classes such as food, mouse, truck, building, person, potted plant, boat, train, table, book, bench, dinner table, motorbike, car, bus. This is due to several different factors. Some objects – e.g., truck, building, person, boat, and train – have a large variety of image appearance. Other objects – e.g., table, dinner table – are frequently partially occluded. Other objects – e.g., plotted plant and book – are often small and poorly lit.

The weak performance on the Track, Keyboard, Lights, Plate, Bottle and Bike classes is also due to several factors. Some objects – e.g., light, plate, keyboard – occur in small sizes. Other objects – e.g., bikes and track – have very heterogeneous appearance. Others, like bottle, are both small and often poorly lit.

Figs. (1a, 7) show typical examples of our results. Note that some objects may be described well by a single segment. Others are often well-described by three segments (e.g., person, car). Even some objects in the weak performance class – e.g.. bike – are sometimes covered fairly well by a small number of segments.

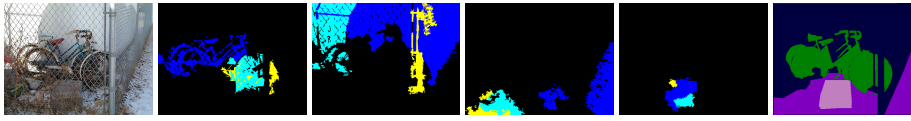


Fig. 7: Left-to-right and top-to-bottom: Original image, top three segments for bike, wall, snow, rock, and ground truth. Note that the segments are good even for object classes that perform poorly overall (e.g., bike).

4.3 Segment combinations as candidate regions

In this section we evaluate the coverage of objects by combinations of our selected-segments set $\mathcal{S} \subseteq \mathcal{S}_{\mathcal{H}}$. From the results of our previous section we conclude that three segments are sufficient to give good coverage. Hence we allow combinations of up to three selected-segments. We obtain an average of 116[1] selected-segments per image after selection and from these we make 721 combinations. This number is the average number of candidate region proposals that we output per image. We generate the candidates by combining neighbors from different levels of the hierarchy, with a maximum difference of 5 levels. Big selected-segments (more than 15,000 pixels) are only combined with selected-segments that are no more than five times smaller. We suppress repeated or too similar proposals.

We evaluate the generated candidate regions with the ground truth on our dataset. We compare to three state-of-the-art methods. In [4], the segment combinations generated provide an average of 1,322 candidate regions for each image. These are generated by taking combinations of the 150 segments (on average) that their hierarchical segmentation approach outputs for each image. Their method is more sophisticated than

[1] In table (1) we relax our selection criterion to obtain a set of 155 selected-segments. This enables us to compare with CPMC [7] who output a set of similar size.

	sky	water	road	*bed*	grass	*cat**	mountain	*tv**	floor	*clothes*	wall
CPMC	65.1	65.4	56.3	65.4	56.5	85.5	55.2	76.8	54.5	58.5	51.7
UCM-combs	90.4	83.8	82.4	81.6	79.3	84.4	75.7	84.5	76.7	76.0	75.2
Sel. search	88.9	77.7	73.4	66.5	74.5	68.7	72.5	65.1	69.2	67.8	64.4
Our sel-segs	86.2	70.6	73.4	57.6	65.2	62.6	67.8	59.0	68.0	63.8	65.2
Our CR combs	90.4	83.2	81.0	79.4	78.1	78.1	76.7	76.7	76.5	75.7	74.5
	*sofa**	rock	*dog**	ground	tree	*bird**	*train**	*plane**	*sheep**	*car**	*cow**
CPMC	64.2	61.1	82.6	55.0	54.5	74.8	78.2	78.6	73.7	72.1	79.9
UCM-combs	76.4	75.6	80.7	76.5	76.5	76.9	75.7	78.4	74.7	73.8	79.4
Sel. search	70.0	66.0	68.9	67.2	68.8	67.3	59.9	63.5	64.6	62.0	60.4
Our sel-segs	64.6	57.3	58.9	63.2	59.2	63.2	53.6	59.0	59.6	55.4	56.5
Our CR combs	74.4	73.7	73.6	73.6	73.5	72.2	72.0	72.0	71.7	71.5	71.1
	*bus**	*cabinet*	snow	sidewalk	building	*bottle**	*horse**	wood	*window*	*pc*	*door*
CPMC	78.9	54.9	60.0	53.0	51.6	74.9	77.5	49.4	50.9	70.7	40.0
UCM-combs	74.7	75.0	71.9	81.0	72.2	78.7	78.1	75.1	70.3	72.1	68.2
Sel. search	58.1	63.0	66.1	71.9	62.8	60.6	61.6	70.6	62.4	54.8	65.3
Our sel-segs	51.3	57.9	58.1	65.8	56.6	57.7	57.4	60.5	58.2	52.1	58.9
Our CR combs	71.1	71.0	70.7	70.6	70.5	69.7	69.6	69.2	69.1	67.6	67.2
	flower	*keyboard*	*d. table**	*truck*	*motorbike**	*boat**	*shelves*	*person**	*pot. plant**	*chair**	ceiling
CPMC	51.0	46.8	62.2	61.8	74.2	71.2	52.6	67.6	63.9	51.7	22.4
UCM-combs	64.2	75.0	73.5	70.4	69.5	70.4	65.5	73.3	73.1	71.3	57.0
Sel. search	65.8	73.1	60.8	64.5	60.4	66.3	63.3	56.9	54.6	56.9	61.2
Our sel-segs	53.7	61.6	52.6	52.9	54.3	51.1	56.7	51.7	54.4	49.1	53.3
Our CR combs	67.1	66.5	66.2	66.2	66.2	66.1	65.6	64.5	64.5	63.5	61.9
	table	*book*	fence	*bag*	*bench*	*bicycle**	*food*	track	*plate*	*sign*	*cup*
CPMC	48.9	36.3	42.3	53.2	36.5	64.4	44.9	42.3	44.5	24.6	40.9
UCM-combs	64.1	67.5	61.3	72.3	63.3	63.3	67.1	54.5	57.2	44.6	65.7
Sel. search	52.6	62.8	53.4	58.5	46.7	48.6	67.2	49.4	51.3	58.9	56.6
Our sel-segs	52.4	49.5	49.1	51.6	44.3	40.8	47.4	44.0	40.5	40.5	45.9
Our CR combs	61.4	59.9	57.8	56.3	56.2	55.0	52.9	50.9	46.2	44.5	44.3

	light	*mouse*	**all IoU**	recall	# cands.	time
CPMC	8.3	12.6	59.6	57.6%	**150**	250s
UCM-combs	27.8	52.3	**77.0**	**80.0%**	1322	850s
Sel. search	38.9	56.8	67.8	66.1%	2100	**4s**
Our sel-segs	40.8	31.3	62.4	55.6%	**155**	**4s**
Our CR combs	37.6	26.8	**74.0**	**70.3%**	721	**4s**

Table 1: Region-based IoU (in %) comparison. CPMC [7], UCM [4], Sel. Search [22], our selected-segments, and our CR – candidate regions. The last four columns show in bold the two best performances. The 20 Pascal VOC classes are marked with (*). Foreground classes are in italic script, background classes are normal script.

ours and we observe that they tend to get larger and less homogeneous segments than we do. The use of gPb makes their method significantly slower than ours (but it also focuses on image segmentation, which we do not attempt). In table (1) we refer to their segments as UCM-combs and to our candidate regions as CR-combs.

Another method we compare to is the Constrained Parametric Min-Cuts (CPMC) [7]. Their method is designed for foreground objects, which explains its better performance on foreground objects. Their overall performance on the 57 classes is lower than our performance, both for our 721 candidate regions (CR-combs), and for our 155 selected-segments (sel-segs). We show the latter for a fair comparison with CPMC's 150 candidates. It is interesting that CPMC is much better on the 20 Pascal VOC objects, marked with (*) in table (1). However, our sel-segs and CR-combs are better than CPMC on most of the 20 newly labeled foreground classes (e.g, bench, book, sign, etc), as well as on the 17 background classes. This suggests that CPMC may be overtuned for those 20 classes, degrading its performance for all other foreground classes.

Finally, we also compare to the Selective Search [22] method because it is competitive in terms of speed. It is optimized for bounding box candidates and performs better on this task: 89.7% box-based IoU and 95.7% box-based recall, compared to 89.5% box-based IoU and 90.9% box-based recall of our method. The recall criterion is the one [22] used: an object is found if its IoU is larger than 50%. However, our method outperforms theirs on the region candidates task (74.0% compared to 67.8% IoU), with less than half the number of proposals. See table (1) for detailed per-class region-based IoU and recall.

The 57-class evaluation is performed on the intersection of our labeled dataset (VOC 2010 detection trainval) and the set of segments provided by [4] (VOC 2011 segmentation trainval), which makes a total of 1288 images. The results are shown in table (1). The objects are sorted by the performance of our CR method. Our performance is lower than [4] but comparable: 74% IoU versus 77% for [4]. But we achieve it with nearly half the number of combinations – 721 compared to 1,322 – and with a simpler and faster algorithm (4s per image in Matlab).

5 Conclusions and future work

We propose a novel and simple method for obtaining candidate regions for objects and background regions. The algorithm is based on simple image cues and is very fast (4s in Matlab) making it applicable to large-scale datasets (e.g., ImageNet).

Our approach has two novelties. Firstly, using the PageRank guiding the hierarchical grouping, which enables global properties of the image to be respected. Secondly, the use of the entropy gain criterion to select maximal homogeneous image segments. The selected-segments form an intermediate-level representation which can be used as a precursor for object detection and classification.

Experiments show that it gives good results on a novel labeling of PASCAL VOC dataset (providing labels for 57 object classes). Firstly, we show that typically one to three segments cover a large part of the object. Secondly, we compare the performance of our candidate regions to the state-of-the-art methods, the best-performing of which is based on UCM. Our performance is comparable to theirs but is obtained using roughly half the number of candidates, by an algorithm which is two orders of magnitude faster. The smaller number of segment combinations is an advantage for a subsequent object classification stage.

It is surprising that such a simple method gives such good results. We plan future work in two directions. Firstly, to see if we can reduce the number of our selected-segments even further by exploring variants of our method including the use of other low-level cues and selection criteria. Secondly, we will start training classifiers on the candidate regions to detect objects and background regions.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. TPAMI 34(11), 2274–2282 (2012)
2. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. TPAMI 34(11), 2189–2202 (2012)
3. Alpert, S., Galun, M., Brandt, A., Basri, R.: Image segmentation by probabilistic bottom-up aggregation and cue integration. TPAMI 34(2), 315–327 (2012)
4. Arbelaez, P., Hariharan, B., Gu, C., Gupta, S., Malik, J.: Semantic segmentation using regions and parts. In: CVPR (2012)
5. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. TPAMI 33(5), 898–916 (2011)
6. Peñalver Benavent, A., Escolano Ruiz, F., Saez, J.M.: Learning Gaussian Mixture Models With Entropy-Based Criteria. TNN 20(11), 1756–1771 (Sep 2009)
7. Carreira, J., Sminchisescu, C.: Cpmc: Automatic object segmentation using constrained parametric min-cuts. TPAMI 34(7), 1312–1328 (2012)
8. Chandler, D.M., Field, D.J.: Estimates of the information content and dimensionality of natural scenes from proximity distributions. J. Opt. Soc. Am. A 24(4), 922–941 (Apr 2007)
9. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR (2009)
10. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV 59(2), 167–181 (Sep 2004)
11. Franceschet, M.: Pagerank: standing on the shoulders of giants. Commun. ACM 54(6), 92–101 (Jun 2011)
12. Galun, M., Sharon, E., Basri, R., Brandt, A.: Texture segmentation by multiscale aggregation of filter responses and shape elements. pp. 716–. ICCV '03 (2003)
13. Kozachenko, L.F., Leonenko, N.N.: Sample estimate of the entropy of a random vector. Probl. Inf. Transm. 23(1), 95–101 (1987)
14. Leonenko, N., Pronzato, L., Savani, V.: A class of Rényi information estimators for multidimensional densities. Ann. Statist. 36(5), 2153–5182 (2008)
15. Lim, J., Zitnick, C.L., Dollár, P.: Sketch tokens: A learned mid-level representation for contour and object detection. In: CVPR (2013)
16. Liu, M.Y., Tuzel, O., Ramalingam, S., Chellappa, R.: Entropy rate superpixel segmentation. In: CVPR. pp. 2097–2104 (2011)
17. Ma, Y., Derksen, H., Hong, W., Wright, J.: Segmentation of multivariate mixed data via lossy coding and compression. TPAMI 29(9) (2007)
18. Maire, M., Arbelaez, P., Fowlkes, C.C., Malik, J.: Using contours to detect and localize junctions in natural images. In: CVPR (2008)
19. Malisiewicz, T., Efros, A.A.: Improving spatial support for objects via multiple segmentations. In: BMVC (2007)
20. Singh, H., Misra, N., Hnizdo, V., Fedorowicz, A., Demchuk, E.: Nearest neighbor estimates of entropy. American J. of Math. and Mgmt. Sci. 23(3–4), 301–321 (2003)
21. Todorovic, S., Ahuja, N.: Region-based hierarchical image matching. IJCV 78(1), 47–66 (Jun 2008)
22. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. International Journal of Computer Vision 104(2), 154–171 (2013)
23. Xu, C., Xiong, C., Corso, J.J.: Streaming hierarchical video segmentation. In: ECCV (2012)