Image Segmentation by Cascaded Region Agglomeration

Zhile Ren
Zhejiang University
jrenzhile@gmail.com

Gregory Shakhnarovich
Toyota Technological Institute at Chicago
greg@ttic.edu

Abstract

We propose a hierarchical segmentation algorithm that starts with a very fine oversegmentation and gradually merges regions using a cascade of boundary classifiers. This approach allows the weights of region and boundary features to adapt to the segmentation scale at which they are applied. The stages of the cascade are trained sequentially, with asymetric loss to maximize boundary recall. On six segmentation data sets, our algorithm achieves best performance under most region-quality measures, and does it with fewer segments than the prior work. Our algorithm is also highly competitive in a dense oversegmentation (superpixel) regime under boundary-based measures.

1. Introduction

A standard preprocessing step in many recognition tasks today is to partition the input image into a set of *superpixels*: "perceptually meaningful atomic regions" [1]. In a typical vision system these number in the hundreds. Sometimes a coarser partition is used, with only tens (or perhaps just a handful) of regions; in this regime the regions are no longer atomic, but the hope is that they remain "perceptually meaningful", that is, that each region does not straddle boundaries between semantically distinct regions, such as boundaries of an object, or occluding boundaries. Thus, such image partition is often called oversegmentation.

Regions extracted by oversegmentation usually form a representation of an image that is much more compact than the original pixel grid. As long as the oversegmentation indeed does not undersegment any region (i.e., little or no "leakage" of true regions across oversegmentation boundaries), this speeds up reasoning, with little loss of accuracy for the downstream task, such as category-level segmentation [3, 7] or depth estimation [25]. A user of oversegmentation algorithm has two conflicting objectives: On the one hand, make the superpixels as large as possible, hence reducing their number; on the other hand, preserve true boundaries in the image, hence driving the number of superpixels up and their size down. All the state-of-the-art superpixel methods provide a tuning parameter that controls this tradeoff, and the range that seems to work for a variety of applications is 400 to 1000 superpixels.

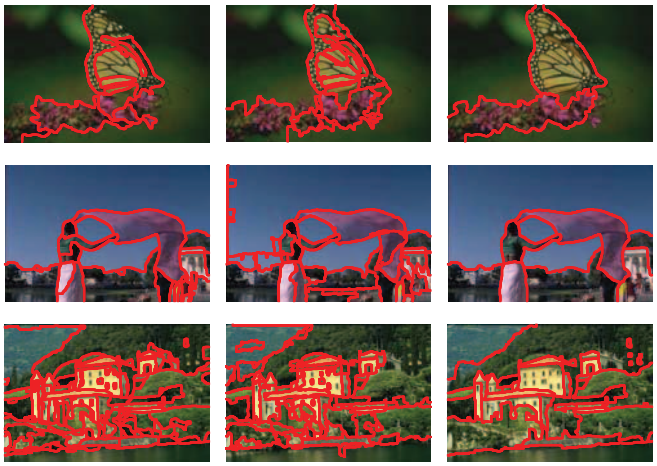

Figure 1. Example segmentations by OWT-UCM (left), Hoiem et al. (middle) and ISCRA (right). For each image, segmentation shown is at the scale optimal for the image (OIS) with each algorithm - i.e., the best that algorithm can do for the image.

Our goal is to re-negotiate this tradeoff, and achieve the same, or better, level of performance as existing methods, but with fewer regions. We pursue the agglomerative clustering approach: starting with a very fine partition into small regions, gradually merge them into larger and larger ones. Fundamental to this is a probabilistic model for grouping regions. We share this basic spirit with some previous work, notably [2, 11], but introduce a number of key innovations. Our method constructs a *cascade* of boundary classifiers that produce increasingly coarse image partitions by merging regions preserved by previous stages. Each stage employs a probabilistic model adapted to the scale at which it operates. These models are trained with scaled asymmetric loss, tuned automatically to optimal precision/recall behavior for the stage. This architecture, that we call Image Segmentation by Cascaded Region Agglomeration (ISCRA), and the learning algorithm for training it are the main contribution of our paper.

1063-6919/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPR.2013.262

We show that across six data sets, performance measures and for a broad range segmentation scale from fine to coarse, the performance of ISCRA is superior to that of current state-of-the-art methods. To our knowledge, it is by far the most comprehensive evaluation of the leading approaches to superpixel extraction, and in general to image segmentation. Some typical examples for ISCRA compared to leading other methods are shown in Figure 1.

2. Background

There is a rich body of work on edge or boundary detection in computer vision, with the state of the art represented by the gPb boundary detector and its variants [2]. However, a boundary map may not correspond to a valid segmentation, since it may not provide closed contours. Below we review related work that produces explicit partition of image into regions. Note that we leave out of our discussion here the extensive recent literature on "semantic segmentation", i.e., category-level labeling of every pixel in the image. In this paper we are only concerned with methods that are agnostic about any category-level reasoning.

For the purpose of our discussion, we will define two segmentation regimes. The *superpixel regime* corresponds to more than 50 segments per image. In this regime the purpose of oversegmentation is mostly to reduce complexity of representation, without sacrificing future segmentation accuracy. Therefore, the natural notion of *scale* for this regime is the number of segments k; typical values of k are in the hundreds for a moderately sized image.

Superpixels, introduced in [17], have become a staple in vision research. Broadly, methods that produce superpixels can be grouped into graph-based [18, 6, 14, 22], clustering of pixels such as SLIC [1] and MeanShift [4], and curve evolution such as Turbopixels [13] and SEEDS[21].

In contrast, the *large segment* regime produces fewer than 100 segments. The appropriate number of segments in this regime depends on the content of the image, and specifying k is not natural. Instead, the precise meaning of scale and the way it is controled varies between segmentation methods, as described below.

In OWT-UCM [2] the oriented watershed transform on gPb boundaries is followed by greedy merging of regions, resulting in weighted boundary map such that thresholding it at any level produces a valid segmentation; the value of the threshold controls the scale. Throughout the merging process, OWT-UCM uses the same set of weights on various features throughout the process, and thus despite the greedy iterative nature of the merging, this is in a sense a single stage process. In contrast, ISCRA uses a large cascade of stages, with weightes learned per stage. We show in Section 5 that this leads to performance better than that of OWT-UCM.

The agglomerative merging segmentation algorithm in [11], like ISCRA, starts with a fine oversegmentation, learns a boundary probability model, applies it to merge regions until the estimated probability of merging is below a threshold. The classifier is then retrained, and applied again; their implementation includes four such stages, therefore defining four segmentation scales. There is a number of differences, however: while in ISCRA we use asymmetric loss and a universal threshold of $\frac{1}{2}$, in [11] the loss is symmetric, but the threshold is tuned in ad-hoc fashion. Consequently, in ISCRA we are able to learn many more stages (60 vs. four), producing a more gradual and accurate merging process. We show empirically that this contributes significantly to performance.

Higher Order Correlation Clustering (HOCC) [12] also starts with fine segmentation, but instead of a greedy merging applies a "single-shot" partition over the superpixel graph. The scale in HOCC is controled by specifying explicitly the number of regions, which may be a disadvantage when the user would like a more adaptive scale definition like in OWT-UCM or in ISCRA.

Finally, a recently proposed method called SCALPEL [24] shares many similarities with our work. In SCALPEL, a region is "grown" by applying a cascade of greedy merging steps to an initial over-segmentation. Similarly to ISCRA, the order of merging in each step is determined by learning weights that reflect importance of features at different scales. However, SCALPEL relies heavily on class-specific shape priors for known object classes, and its objective is to produce a set of class-specific region proposals, that can be used in semantic image labeling. This is in contrast to our work, which is agnostic with respect to categorization of regions. Thus the two methods, while sharing many of the ideas, are not directly comparable.

An important question about any segmentation algorithm is how it handles multiple scales. Specifically, it is often desirable to produce a *hierarchy*, in which regions obtained at a finer scale are necessarily subregions of the regions at a coarser scale. This is the case with OWT-UCM, the agglomeration algorithm of [11], and with ISCRA, but generally not with the graph-based algorithms like [12, 18]. This is also not the case with superpixel algorithms like SLIC, ERS, SEEDS or Turbopixels.

Although the definitions above of the two regimes are somewhat arbitrary, we adopt them for convenience. Still, it seems clear that the objectives in partitioning an image into 1000 segments vs. just ten are different, and it dictates different evaluation protocol for these cases. Some algorithms, including ISCRA, are competitive in both regimes, as demonstrated in Section 5.

3. Problem setup

For an image I_i partitioned into a set of k_i regions $\mathcal{R}_i = \{R_{i,1}, \ldots, R_{i,k_i}\}$, we consider the set $\mathcal{N}(\mathcal{R}_i)$ of all neighboring regions. Every pair $(p,q) \in \mathcal{N}(\mathcal{R}_i)$ is associated with a feature vector $\boldsymbol{\phi}_{p,q}(I_i, \mathcal{R}_i)$; details of this feature representation in our implementation are in Section 5.1.[1]

We are also given ground truth segmentation for a set of training images. A ground truth segmentation is provided as a label map. We do not assume any semantically meaningful labels, and the only information we get from the ground truth is which pixels belong to the same region. We can also have multiple ground truths for an image, for instance provided by g human labelers, as is the case in the Berkeley Segmentation Data Set (BSDS) [16].

The set of ground truths for I allows us to label each region pair in $\mathcal{N}(\mathcal{R}_i)$, as follows. Given a single ground truth map G_i we assign every $R_{i,p}$ to the region in G_i with the highest overlap with $R_{i,p}$. Then, if $R_{i,p}$ and $R_{i,q}$ are assigned to the same region, we set $y^i_{pq} = 1$, otherwise $y^i_{pq} = 0$. When multiple ground truths are available, we set y^i_{pq} to the average value of the labels assigned under each of the ground truths. This yields y^i_{pq} between 0 and 1, measuring the fraction of humans who thought $R_{i,p}$ and $R_{i,q}$ belong in the same region (and thus, intuitively, reflecting the perceptual strength of the boundary).

Stacking the pairs of regions in training images $I_1, \ldots, I_N$ and their labels and simplifying the indexing, we get a set $\{\langle \boldsymbol{\phi}_i, y_i \rangle\}_{i=1}^n$. This induces a prediction problem: given an image I and initial segmentation $\mathcal{R}$, estimate the conditional posterior *probability of grouping* $Pg(p,q;I,\mathcal{R}) \triangleq P\left(y_{pq} = 1 \,|\, \boldsymbol{\phi}_{pq}(I,\mathcal{R})\right)$ for every pair $(p,q) \in \mathcal{N}(\mathcal{R})$. If $Pg(p,q;I,\mathcal{R}) > \frac{1}{2}$ we must merge R_p and R_q and eliminate the boundary between them, otherwise we preserve the boundary.

3.1. Greedy merging of regions

While it is possible to compute the predictions for all pairs in $\mathcal{N}(\mathcal{R}_i)$ at once, the resulting set of predictions may suffer from inconsistencies (lack of transitivity in the predicted labels). One could tackle this problem leading to a fairly complex optimization problem, e.g., as in [12]. Instead, we pursue a simpler approach: greedy merging of regions. In each iteration, we merge the pair of regions with the highest Pg update the features to reflect this merge, and repeat, until no pair of current regions has $Pg > \frac{1}{2}$. Since we started with a valid segmentation and coarsened it, we remain with a valid segmentation $\mathcal{R}'$. This is summarized in Algorithm 1. Note that the update step will affect all regions that were neighbors of either a or b before the merge, but will not affect any other regions.

[1] For brevity, we drop dependence on image index i from notation when this doesn't lead to confusion.

Algorithm 1: Greedy merging MERGE$(I, \mathcal{R}, \mathbf{w})$

Given: image I, regions $\mathcal{R}$, weights $\mathbf{w}$
foreach $(p,q) \in \mathcal{N}(\mathcal{R})$ **do** $P_{p,q} = Pg(\boldsymbol{\phi}_{pq}(I,\mathcal{R}); \mathbf{w})$
initialize $\mathcal{R}' = \mathcal{R}$
while $\max_{p,q} P_{p,q} \geq \frac{1}{2}$ **do**
  $(a,b) = \operatorname{argmax}_{p,q} P_{p,q}$
  merge $R_a \leftarrow R_a \cup R_b$ in $\mathcal{R}'$
  foreach j *s.t.* $(a,j) \in \mathcal{N}(\mathcal{R}')$ **do**
    update $\boldsymbol{\phi}_{a,j}(I,\mathcal{R}')$
    update $P_{a,j} = Pg(\boldsymbol{\phi}_{aj}(I,\mathcal{R}'); \mathbf{w})$
Return: $\mathcal{R}'$

3.2. Training with scaled loss

We model Pg with linear logistic regression:

$$Pg(p,q;I,\mathcal{R}) \;=\; 1/\left(1 + \exp\left\{-\mathbf{w}^T \boldsymbol{\phi}_{pq}(I,\mathcal{R})\right\}\right).$$

The model is usually trained by minimization of log-loss, averaged over examples (which here means averaged over region pairs in the training set). However, naive use of log-loss fails here, because of the symmetry with which it penalizes two types of mistakes: over- and underestimating Pg. Given a typical image with 1000 superpixels in $\mathcal{R}$, only a small fraction of pairs in $\mathcal{N}(\mathcal{R})$ will be true negatives, since most neighboring superpixels should be grouped together. A similar problem was identified in [12], leading to a modification of Hamming loss.

Our solution is also to modify the loss: the cost of the false positive (wrongly merged pair of regions) is multiplied by some α. Furthermore, we scale the loss value for every pair of regions by the length L_i, in pixels, of the boundary between the regions. This scaling reflects the higher loss from "erasing" a long true boundary than a short one:

$$\mathbf{w}^*(\alpha) \;=\; \operatorname*{argmin}_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} L_i \Big[y_i \log Pg(\boldsymbol{\phi}_i; \mathbf{w}) + \alpha(1-y_i) \log(1 - Pg(\boldsymbol{\phi}_i; \mathbf{w})) \Big] \tag{1}$$

The scaled loss in (1) is convex and optimizable in the same way as the "normal" logistic regression.

Our choice for α is driven by the following intuition: we would like to *preserve* the boundary recall of the starting segmentation as much as possible, while merging as many regions as possible. The boundary recall REC$(\mathcal{R}, G)$ is defined as the fraction of boundary pixels in ground truth G recovered by the predicted boundaries in $\widehat{\mathcal{R}}$, averaged over a set of images.

The recall tends[2] to behave as a monotonically increasing function of α. Suppose the recall of the initial segmentation is r; then, we want the lowest value of α for which

[2] but is not guaranteed to behave so, unless $I_{train} = I_{tune}$; empirically however we always observed this behavior.

the recall does not drop below $r - \rho$ for some small ρ. This dictates an efficient algorithm for optimizing α: start with bracketing α between 0 and some large number, and perform binary search. We use two subsets of images: $\mathcal{I}_{train}$ is used to learn $\mathbf{w}^*(\alpha)$ in each iteration of the binary search as per 1, while $\mathcal{I}_{tune}$ is used to evaluate average recall obtained with $\mathbf{w}^*(\alpha)$ on images in $\mathcal{I}_{tune}$.

4. Cascaded Region Agglomeration

When the model is trained with the scaled loss (1), with α optimized to limit recall drop on tuning set, the merging usually stops early, with many remaining unmerged regions. This is in part due to the very "cautious" model learned with the scaled loss, and in part to the fact that once many of the regions in the initial segmentation are merged, and the features of their surviving neighbors are updated, the distribution of the features no longer matches the one on which we trained Pg. Color and texture histograms tend to become less sparse; shape may become less convex; features that were useless for very small regions (e.g., counts of visual words) may become useful for larger regions, etc. This observation leads to a simple idea: re-train the model on the new, larger regions. The second model merges some more segments, but then it also stops. We can then train a third model, etc., as described below.

Algorithm 2: Cascaded Segmentation

Given: Image I, initial $\mathcal{R}$,weights $\mathbf{w}_1, \ldots, \mathbf{w}_T$
$\mathcal{R}_0 \leftarrow \mathcal{R}$
for $t = 1$ **to** T **do** $R_t \leftarrow \text{MERGE}(I, \mathcal{R}_{t-1}, \mathbf{w}_t)$
Return: $\mathcal{R}_T$

Training the cascade (Algorithm 3) is similar to the training of cascaded classifiers elsewhere, e.g., in the Viola-Jones face detector [23]. It is also similar in spirit to the re-training of pixel merge likelihood in [11]. One important difference from these is that we use asymmetric loss, rather than tune the threshold on classification. This enables us to train a deeper cascade, and helps performance as we show in Section 5.

Algorithm 3: Training a cascade

Given: $\{I_i, \mathcal{R}_i, G_i\}_{i=1}^N,\ \rho > 0,\ T$
for $t = 1$ **to** T **do**
  sample image subsets $\mathcal{I}_{train}, \mathcal{I}_{tune}$,
    s.t. $\mathcal{I}_{tune} \cap \mathcal{I}_{train} = \varnothing$
  Find α_t^* with binary search, using $\mathcal{I}_{train}, \mathcal{I}_{tune}, \rho$
  $\mathbf{w}_t \leftarrow \mathbf{w}^*(\alpha_t^*)$ by Eq. (1)
  foreach $i \in \{1, \ldots, N\}$ **do**
    merge $R_i \leftarrow \text{MERGE}(I_i, \mathcal{R}_i, \mathbf{w}_t)$
Return: $\mathbf{w}_1, \ldots, \mathbf{w}_T$

At each stage of Algorithm 3, $\mathcal{I}_{train}$ and $\mathcal{I}_{tune}$ are mutually exclusive, to prevent overfitting of α. Furthermore, at each stage these two sets are sampled independently, so that empirical distribution of features on with stage t is a more robust estimate of the distribution of new data. Finally, note that after a few stages most of the boundaries from earlier stages are no longer active (due to merging of their constituent regions) and so reusing the same images carries much less risk of overfitting.

To summarize: ISCRA starts with an initial set of small superpixels, and propagates them through a series of stages. At each stage some of the regions are merged using the model learned for that stage, and the next stage receives the resulting coarsened segmentation as its input.

Once the merging stops, we can "backtrack" and report the segmentation at any point along the merging process. This allows us to control the scale either by specifying the desired number of segments (appropriate for the superpixel regime) or by specifying the number of stages to run, which is the natural definition of scale for ISCRA. We can also compute the boundary map that reflect the scale at which regions are merged: if there are T stages in ISCRA, then for every boundary pixel in the initial segmentation, the value of the boundary map will be t/T if it was merged after t stages. Pixels that were not on the boundaries of initial superpixels will have values zero, and pixels that survived the last stage will have value 1. Examples of ISCRA segmentations at multiple scales, as well as the hierarchical boundary map, are illustrated in Figure 2(right); more examples are available in supplementary materials.

5. Experiments

Our experiments were aimed at two goals: (i) compare ISCRA to other methods in both superpixel and large region regimes; (ii) evaluate effect of various design choices on performance. Below we describe a few remaining implementation details, the experimental setup and the results.

5.1. Features and training

The features we use can be grouped into three sets:

Appearance features that measure difference in the "content" of the two regions. These include

- **Color**: The χ^2 difference of color histograms, computed for each channel in L*a*b color space, with 32 bins per channel. This yields 3 dimensions in $\boldsymbol{\phi}$.
- **Texture**: The χ^2 difference of two segments when representing each image using 32 textons (1 dimension).
- **Geometric Context**: For each of seven geometric context labels [10], compute χ^2 difference between histograms of values for that label within the regions, with 32 bins (7 dimensions).

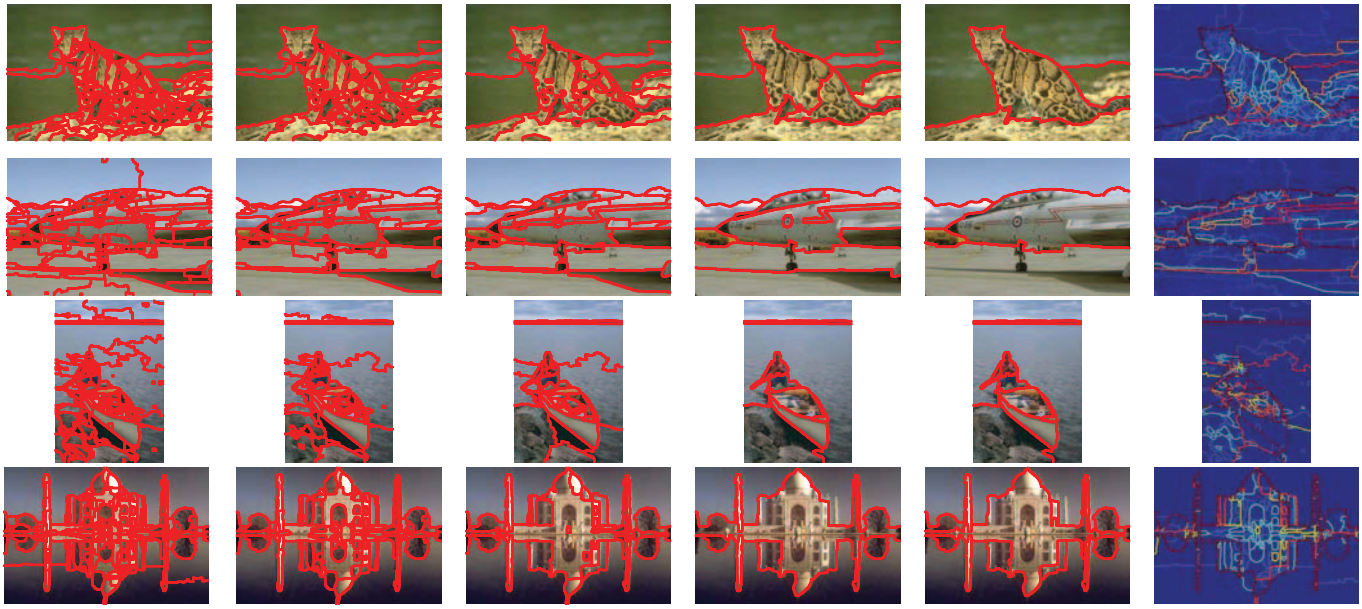

Figure 2. Examples of the cascaded merging by ISCRA on BSDS test images. From left: results after merging from ≈1000 superpixels down to 250,125,50,10, the segmentation with ISCRA at the optimal scale in hindsight for the image (OIS), and the boundary strength map.

- **SIFT**: χ^2 difference between histogram of SIFT dictionary of 30 words computed for the image. We extract SIFT descriptors at a dense grid of 8 pixels, with two patch sizes: 8 and 18 pixels, and do this for both the gray level image and for the a and b color channels (6 dimensions).

Shape features include region properties, area, perimeter, area of convex hull, eccentricity, centroid and bounding box, Euler number etc. (19 dimensions).

Boundary features measure properties of the boundary between the regions. We compute these as the average (per pixel) values of gPb and of the OWT-UCM (2 dimensions).

With the addition of constant bias term for each stage, we have in $\mathbf{w}_t$ for stage t 39 weights. Note that α_t is an internal hyper-parameter in training; it serves to select the $\mathbf{w}_t$, but needs not be reported to the user of the algorithm.

We trained 60 stages ISCRA on the 200 images in BSDS300 training set, using unregularized logistic regression at each stage (see discussion in Section 4 regarding overfitting). In each stage, 120 images were used as $\mathcal{I}_{train}$ and a subset of the rest as $\mathcal{I}_{tune}$; we set tolerance on recall drop, used in search for optimal α, to $\rho = 0.01$. We also constrained merging in Alg. 1 to prevent repeated merging involving the same region at the same stage; empirically this improves performance, since it keeps the training data distribution from changing too much within the stage. As initial segmentation for every image (train or test) we took the finest scale of OWT-UCM obtained for that image. Training took approximately 5 hours on a quad-core machine; most of the time was spent in search for optimal α. Note that OWT-UCM is trained on the same data, while method of Hoiem et al is trained on a different data set. We then apply the trained models to all the data sets without any retraining.

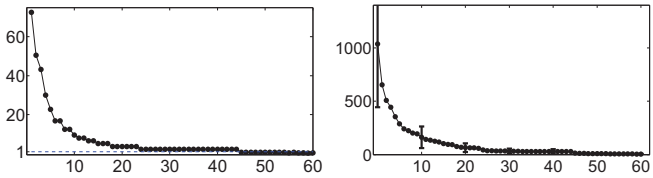

Figure 3. (left) Values of α selected in training. (right) average number of superpixels surviving each stage, over all images in all test sets (with standard deviation bars).

Figure 3(left) shows the trajectory of optimal value of α through stages. As might be expected, this value decreases dramatically with the average number of segments surviving the preceding stages (shown in Figure 3(right) for the test sets). In final stages $\alpha < 1$, as at that point in many of the images there are more positive than negative examples (i.e., merging is more likely to hurt than help).

5.2. Data sets, methods, and measures

Our experiments covered six data sets:

1. **BSDS300** [16]:, 200 training images (on which ISCRA was trained) and 100 test images. For each image 5-10 ground truth segmentations are given. There are on average 19.6 ground truth segments per image.
2. **BSDS500** [2]: extended version of BSDS300. avg. 22 ground truth segments.

3. **SBD**: Stanford Background Dataset [9], 715 outdoor-scene images. The ground truth we are using is "layers"– labeling of distinct image regions; avg. 20.2 segments.
4. **MSRC** [19]: 591 images and 23 object classes. For ground truth images, we use the cleaner and precise version of [15]; avg. 4.6 segments.
5. **VOC2012**: validation set for segmentation challenge (comp5) Pascal VOC 2012 [5]. 1499 images and 21 classes; avg. 3.4 segments.
6. **NYU Depth** [20]: We used the subset of the 1449 RGB images for which dense instance-level labeling is provided; avg. 25.2 segments.

Two measures are sensible for any segmentation scale: boundary precision vs. recall, and achievable segmentation accuracy (ASA). The latter measures, given a labeled ground truth segmentation,[3] the average per pixel accuracy of optimal (in hindsight) labeling of the hypothesized regions. See [14] for more details.

Superpixels In superpixel regime, we compare ISCRA to SLIC, ERS, OWT-UCM, and Hoiem et al. SLIC and ERS have been shown in [1, 14] to outperform other methods, including MeanShift [4], Turbopixels [13], graph-based methods [18, 6, 22], and others [17, 8], on at least some of the data sets, and we do not compare to those methods.

Since individual superpixels tend to be much smaller than most ground truth regions, region-based measures make little sense in this regime. Following the practice in the literature, we focus on a boundary-based measures: precision/recall, and undersegmentation error (measuring amount of pixels "leaking" across true region boundaries) as well as ASA.

Large segment In large segment regime, we compared ISCRA to OWT-UCM, Hoiem et al and to HOCC, the latter only on SBD and BSDS300, for which the authors of [12] report results.[4] Since HOCC was dominated by other methods we do not include detailed results in the tables.

Since pixel-wise errors become fairly large in this regime, the boundary-based measures relevant for the superpixel regime are no longer sensible. Instead, we focus on region-based measures, listed below. See [2] for detailed definitions.

- Segmentation covering, measuring average per-pixel overlap between segments in ground truth (GT) and the proposed segmentation.
- Probabilistic Rand Index (PRI), measuring pairwise compatibility of pixel assignments GT and the proposed segmentation.
- Variation of Information (VOI), measuring relative entropy of the GT and the proposed segmentation.

With the exception of [12] the methods involved produce hierarchical segmentation, which can be used to extract a specific set of regions by specifying a scale value. Following [2] we report the results obtained by optimizing the scale in two ways: jointly for the entire training data set (ODS), and separately for each *test* image (OIS). The latter is a kind of oracle measuring the best achievable accuracy of any labeling adhering to the predicted segmentation regions. Finally, since for some of the measures there is a trade-off between performance and number of segments, we report for each ODS/OIS value the average number of segments extracted at that scale by the method in question.

5.3. Results on benchmarks

Precision-recall curves in Figure 5 show that no single method dominates the others, however ISCRA is comptetitive for the top spot with OWT-UCM. Since only partial ground truth boundaries VOC2012 and MSRC are available, we omit precision/recall curves for those data sets.

Figure 6 shows that for most data sets, ISCRA achieves the lowest under-segmentation error for a broad range of scales, typically for 400 superpixels and fewer. For very fine oversegmentations, with more than 400 superpixels, ERS tends to perform best.

Results for region measures relevant for the large segment regime are summarized in Tables 1 through 6. Across data sets ISCRA clearly dominates other methods on covering and VOI, and is on par with OWT-UCM on PRI. While ISCRA achieves better or equal results in most combinations of data set/measure/scale choice, it typically does that with many fewer segments than other methods (see superscripts in the ODS columns), and the number of segments it obtains is closer to that in the ground truth.

A similar effect is seen in the analysis of ASA performance. Figure 4 shows, for ASA between 0.85 and 0.95, the average number of segments required to get to that level on each data set. Arguably, ASA values below 0.85 become less interesting since they handicap any semantic labeling algorithm that might use the segmentation. Across the board (with the exception of NYU) ISCRA achieves the same level of ASA with significantly fewer segments than other methods; most dramatically for ASA=0.95.

We have evaluated some of the design choices in ISCRA; due to lack of space we only state the conclusions here, and the detailed results are found in the supplementary material. To assess the role played by different features, we trained versions of ISCRA without various subsets of the features; removal of every subset indeed hurts performance. We also trained a variant of ISCRA in which $\alpha = 1$ for all stages, and

[3] In BSDS data sets no semantic labels are provided, and we assign each ground truth region its own label.

[4] [12] also includes results on a subset of MSRC, not comparable to our evaluation on the entire data set.

the threshold on Pg used to stop merging is trained. This variant, too, performs less well than ISCRA.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	0.65	0.59^{23}	1.54	1.66^{10}	0.85	$\mathbf{0.81}^{50}$
Hoiem[11]	0.59	0.55^{17}	1.70	1.83^{10}	0.82	0.79^{17}
ISCRA	**0.66**	$\mathbf{0.60}^{12}$	**1.40**	$\mathbf{1.61}^{8}$	**0.86**	$\mathbf{0.81}^{16}$

Table 1. Results for large segment regime, **BSDS300**. OIS: optimal scale per (test) image, ODS: optimal scale for entire test data set. Superscripts: the average number of segments at the optimal scale for each method/measure. Best results are shown in bold.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	0.64	$\mathbf{0.59}^{18}$	1.49	1.69^{13}	**0.85**	$\mathbf{0.83}^{59}$
Hoiem[11]	0.60	0.56^{18}	1.66	1.78^{11}	0.84	0.81^{18}
ISCRA	**0.66**	$\mathbf{0.59}^{24}$	**1.42**	$\mathbf{1.60}^{10}$	**0.85**	0.82^{24}

Table 2. Large segment regime, **BSDS500**. See caption of Table 1.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	0.74	0.64^{6}	1.05	1.30^{3}	**0.85**	0.78^{12}
Hoiem[11]	0.67	0.65^{7}	1.34	1.37^{7}	0.80	0.77^{8}
ISCRA	**0.75**	$\mathbf{0.67}^{4}$	**1.02**	$\mathbf{1.18}^{3}$	**0.85**	0.77^{14}

Table 3. Large segment regime, **MSRC**. See caption of Table 1.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	**0.58**	0.55^{7}	1.70	1.75^{7}	**0.63**	$\mathbf{0.60}^{7}$
Hoiem[11]	0.56	0.54^{7}	1.72	1.73^{7}	0.61	0.59^{7}
ISCRA	0.57	$\mathbf{0.56}^{5}$	**1.64**	$\mathbf{1.65}^{5}$	0.62	$\mathbf{0.60}^{5}$

Table 4. Large segment regime, **VOC2012**. See caption of Table 1.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	0.64	0.58^{24}	1.63	1.88^{11}	0.89	0.86^{41}
[11]	0.67	$\mathbf{0.62}^{20}$	1.52	1.64^{12}	**0.90**	$\mathbf{0.87}^{20}$
ISCRA	**0.68**	$\mathbf{0.62}^{21}$	**1.50**	$\mathbf{1.73}^{15}$	**0.90**	$\mathbf{0.87}^{36}$

Table 5. Large segment regime, **SBD**. See caption of Table 1.

Method	Covering		VOI		PRI	
	OIS	ODS	OIS	ODS	OIS	ODS
UCM[2]	0.51	0.46^{141}	2.33	2.52^{34}	**0.90**	0.88^{330}
Hoiem[11]	0.51	0.48^{87}	2.36	2.50^{28}	0.89	0.88^{194}
ISCRA	**0.54**	$\mathbf{0.50}^{62}$	**2.21**	$\mathbf{2.34}^{20}$	**0.90**	$\mathbf{0.89}^{97}$

Table 6. Large segment regime, **NYU**. See caption of Table 1.

6. Conclusions

We present ISCRA: Image Segmentation by Cascaded Region Agglomeration. ISCRA consists of a cascade of probabilistic models that predict the probability of grouping neighboring regions. It is trained in sequence, allowing adaptation of feature weights to increasing segmentation scale; when applied on an image it produces a hierarchical segmentation, allowing the user to directly control the scale and the number of resulting regions. In experimental comparison on six data sets, ISCRA is a clear winner in region-based measures. It also is competitive in boundary-based measures in the superpixel regime, obtaining best results for part of the range for some data sets. ISCRA tends to achieve these results with fewer segments per image than other methods, making it potentially appealing for use as preprocessing step for semantic segmentation and other high-level perception tasks.

As the experiments show, a major limitation on ISCRA is its dependence on the initial segmentation. For instance, in our experiments here, it can not obtain ASA or boundary recall values above those of the finest scale of OWT-UCM. We plan to investigate initialization methods that combine fine-scale segmentations from multiple algorithms. We also are working on designing additional features that will improve accuracy of Pg estimated in later stages of ISCRA.

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. *IEEE TPAMI*, 2012.

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE TPAMI*, 33(5), 2011.

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

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE TPAMI*, 24(5), 2002.

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012). http://www.pascal-network.org/challenges/VOC/voc2012.

[6] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2), 2004.

[7] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *ICCV*, 2009.

[8] M. Galun, E. Sharon, R. Basri, and A. Brandt. Texture segmentation by multiscale aggregation of filter responses and shape elements. In *ICCV*, 2003.

[9] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009.

[10] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*. IEEE, 2005.

[11] D. Hoiem, A. Efros, and M. Hebert. Recovering occlusion boundaries from an image. *IJCV*, 91(3), 2011.

[12] S. Kim, S. Nowozin, P. Kohli, and C. D. Yoo. Higher-order correlation clustering for image segmentation. In *NIPS*, 2011.

[13] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *IEEE TPAMI*, 31(12), 2009.

[14] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa. Entropy rate superpixel segmentation. In *CVPR*, 2011.

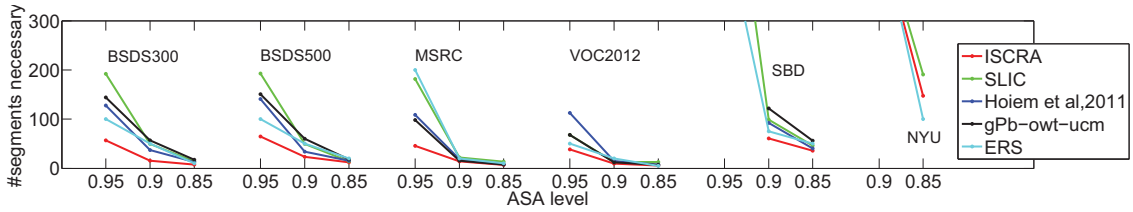

Figure 4. Average number of segments to achieve desired ASA.

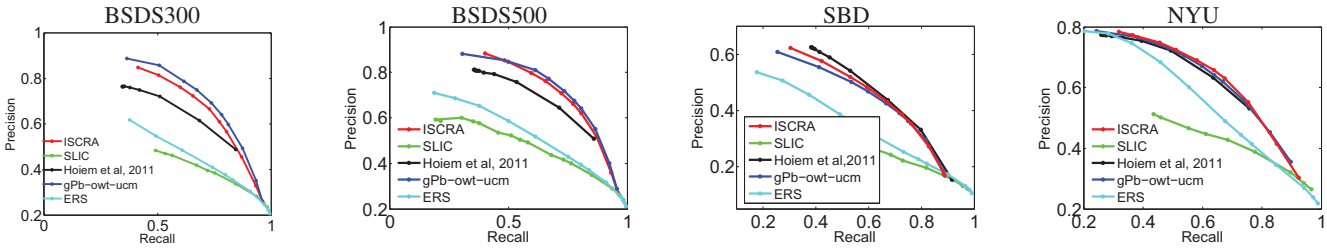

Figure 5. Boundary precision-recall curves. Top right corner is ideal.

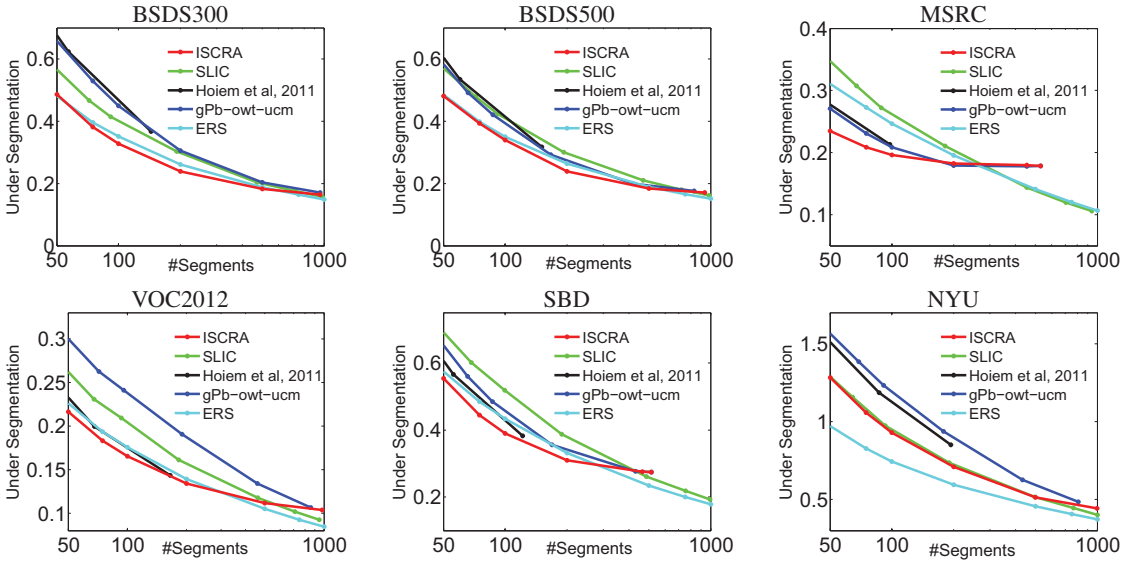

Figure 6. Under-segmentation in superpixel regime. Lower values are better.

[15] T. Malisiewicz and A. A. Efros. Improving spatial support for objects via multiple segmentations. In *BMVC*, September 2007.

[16] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, volume 2, July 2001.

[17] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*. IEEE, 2003.

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 22(8), 2000.

[19] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation. In *ECCV*, 2006.

[20] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[21] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool. SEEDS: Superpixels Extracted via Energy-Driven Sampling. In *ECCV*, 2012.

[22] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. *ECCV*, 2010.

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

[24] D. Weiss and B. Taskar. SCALPEL: Segmentation CAscades with Localized Priors and Efficient Learning. In *CVPR*, 2013.

[25] K. Yamaguchi, T. Hazan, D. McAllester, and R. Urtasun. Continuous markov random fields for robust stereo estimation. In *ECCV*, 2012.