HFS: Hierarchical Feature Selection for Efficient Image Segmentation

Ming-Ming Cheng[1⋆], Yun Liu[1⋆], Qibin Hou[1], Jiawang Bian[1], Philip Torr[2], Shi-Min Hu[3], and Zhuowen Tu[4]

[1]CCCE&CS, Nankai University, [2]Oxford University, [3]Tsinghua University, [4]UCSD
cmm@nankai.edu.cn

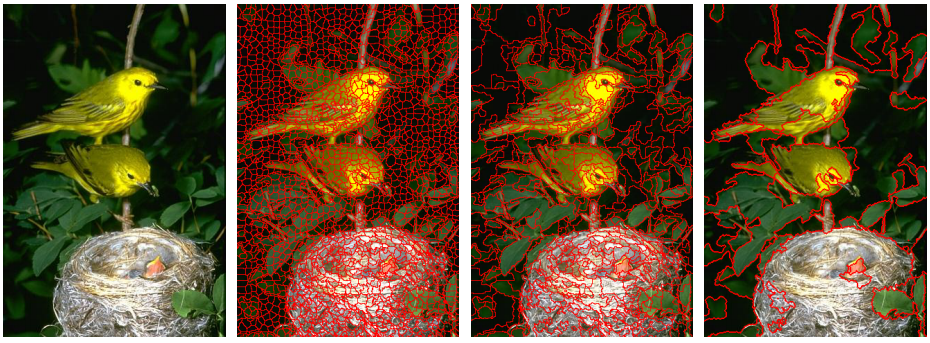
Abstract. In this paper, we propose a real-time system, Hierarchical Feature Selection (HFS), that performs image segmentation at a speed of 50 frames-per-second. We make an attempt to improve the performance of previous image segmentation systems by focusing on two aspects: (1) a careful system implementation on modern GPUs for efficient feature computation; and (2) an effective hierarchical feature selection and fusion strategy with learning. Compared with classic segmentation algorithms, our system demonstrates its particular advantage in speed, with comparable results in segmentation quality. Adopting HFS in applications like salient object detection and object proposal generation results in a significant performance boost. Our proposed HFS system (will be open-sourced) can be used in a variety computer vision tasks that are built on top of image segmentation and superpixel extraction.

Keywords: Image segmentation, superpixel, grouping

1 Introduction

Image segmentation is considered as a main challenge in computer vision that has been extensively studied in the past. After decades of research, a consensus nonetheless exists among researchers in the field that accurate segments, either as large regions or as small superpixels, serve as an effective input representation for middle-level and high-level vision tasks, albeit intrinsically ambiguous. Some typical tasks that have greatly benefited from building on good segmentations include object detection/recognition [24, 25], tracking [44], saliency estimation [10, 22], objectness proposal generation [2, 8, 43], and 3D inference [20]. The reason for this to happen is threefold: i) extracted segments are meaningful units that carry informative features such as shapes, textures, *etc* [21, 26, 39]; ii) the number of segments is often significantly lower than the number pixels in the original image, resulting in a more compact representation with a great speed benefit [10]; iii) the superpixel representation often has an improved coherency and robustness than the raw pixels [37].

⋆ Two joint first authors have made equal contribution to this paper.



(a) input (b) 1^{st} GPU-SLIC (c) 2^{nd} merge (d) 3^{rd} merge

Fig. 1: Sample results from different steps of our methods. The original image (481×321) is from BSDS500 [4] dataset. These image segmentation results are generated at 50fps on GPU.

In the past, several seminal works have emerged as the state-of-the-art systems that have been widely adopted in the field: spectral clustering based normalized cuts approach [38]; efficient feature (color) space mode seeking method, mean-shift algorithm [13]; efficient graph-based image segmentation method [17]; hierarchical region tree with transform contours [4]; and multi-scale normalized cuts algorithm [5]. Among these choices, efficient graph based image segmentation (EGB) and SLIC [1] methods are particularly popular in computer vision and computer graphics [2, 7, 8, 10, 20, 22, 24, 25, 43, 44], due to their great speed advantage.

In this paper, we aim at developing a rapid image segmentation system that produces high quality image segments for real-time computer vision tasks. We propose a hierarchical feature selection framework that learns feature combination in individual stages of a hierarchical structure. Our effort starts with a GPU version of the SLIC method [1, 34], to quickly obtain initial seed regions (superpixels) by performing oversegmentation. Image features are then extracted from the individual seed regions, followed by a feature combination process with a distance metric learnt from the training data. Note that to maintain the efficiency of our system, we only consider those image features that are appropriate for parallel computing, *i.e.* via GPUs. A region merging process is then performed based on the learned distance metric to output a new set of regions for the next level in the hierarchy. Our system then repeats for a few iterations.

The method developed in this paper has its practical importance to a variety of real-time applications by generating high quality image segments (see also Fig. 1) at 50 fps. The performance of our method is quantitatively evaluated in the well-known BSDS500 [4] dataset (see also Sec. 4). As demonstrated in the evaluation results (see also Tab. 2), our method strikes a favorable balance between segmentation quality and computational efficiency when compared with

alternative approaches [1,4,5,17,40]. We will open-source our system to make it publicly available.

2 Related Work

Image segmentation is a fundamental problem in computer vision [30]. We refer readers to the popular BSDS500 [4] benchmark and other recent studies [3, 5, 28, 42] for a comprehensive background discussion. Next, we highlight a few representative methods that are relevant and important to the method proposed here.

A certain degree of attention in the past was given to grouping algorithms that efficiently compute and implement the normalized-cuts algorithm [38]. A multigrid eigenvectors producer is designed [28], enabling substantial speed-up for eigenvector computation. In [42], Taylor *et al.* attempt to reduce the size of eigenvectors using a watershed oversegmentation to achieve the speed of computing eigenvectors in less than half a second. Pont-Tuset *et al.* [5] present an approach to downsample the eigenvectors first, solving them at a reduced size, followed by upsampling the solution to retrieve the structure of the image. Although satisfying segmentation results can be obtained by the above methods, the computational time is still a bottleneck for these spectral clustering based approaches.

Along a different direction, SLIC [1] emerges as one of the most celebrated methods with a good balance between accuracy and speed, and it has been adopted in many applications [6, 9, 23, 41, 49]. In [1], a k-means clustering approach is proposed to initialize cluster centers by sampling pixels at regular grid steps, followed by a labeling procedure in which each pixel is labeled with the index of the cluster center whose search region overlaps with its location.

A graph-based clustering methods presented by Felzenszwalb and Huttenlocher [17] has also been widely used. For an undirected graph with edges measuring the dissimilarity between adjacent pixels, the goal of [17] is to perform a clustering operation such that each region is the minimum spanning tree of the involved pixels. Since it starts with a merging process directly from single pixels with weak color information, the algorithm of [17] is prone to noise. Compared with [17], we instead start our clustering method from oversegmentations that contains more informative features than single pixels. We will discuss in detail in Sec. 3 about our procedure.

Other popular methods for image segmentation include those based on feature learning [35]. These methods demonstrate a good representation power by fusing together features such as brightness, color, and texture properties using discriminative classifiers. Ren *et al.* [35] propose a hierarchical segmentation approach in which a cascade of boundary classifiers are applied to recursively combine regions starting from initial oversegmentations. In this spirit, our work bears certain similarity to [35] where a cascade of classifiers are used for region grouping. Here, we strike the importance of real-time execution by carefully studying regional features that are appropriate for GPU implementation when

combined with [17]. The main contribution of our work is the development of a real-time image segmentation system that is of practical importance to be used in many high-level computer vision tasks.

3 Our Method

In this section, we first introduce the problem formulation and our hierarchical merging algorithm. We then explain the parameter learning and feature extraction procedure, followed by a discussion about design choices behind our method.

3.1 Problem formulation

Given an image I, we partition it into L level segmentations $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_L\}$. Each segmentation $\mathcal{S}_l$ is a decomposition of the image I with K_l regions

$$\mathcal{S}_l = \{R_1^{(l)}, R_2^{(l)}, \ldots, R_{K_l}^{(l)}\}, \tag{1}$$

where l denotes the level index in the hierarchy. We start with the finest segmentation $\mathcal{S}_1$ consisting of a large number of regions, and gradually merge regions from level $\mathcal{S}_l$ to a coarser level $\mathcal{S}_{l+1}$. The coarsest level segmentation thus is composed of fewest regions.

We adopt a graph based approach [17] for the implementation of the region merging process $\mathcal{S}_l \Rightarrow \mathcal{S}_{l+1}$ at each step. Let

$$G_l = (\mathcal{S}_l, \mathcal{A}_l) \tag{2}$$

be an undirected graph, with vertices being a set of regions $\mathcal{S}_l$ as defined above, and edges $(R_i^{(l)}, R_j^{(l)}) \in \mathcal{A}_l$ corresponding to pairs of neighboring vertices. Each edge $(R_i^{(l)}, R_j^{(l)}) \in \mathcal{A}_l$ has a feature vector $\boldsymbol{T}_{i,j}^{(l)}$ (see also in Sec. 3.4), and a corresponding predict score $s_{i,j}^{(l)}$, which is a non-negative measure of the distance between regions $R_i^{(l)}$ and $R_j^{(l)}$.

Based on the above problem definition, our task is then to quickly merge regions to produce coherent segments that best match human annotations, such as those in the BSDS500 [4] benchmark.

3.2 Hierarchical Merging

To achieve high quality and retain top efficiency, we propose to i) iteratively learn how to combine features and update image features after region merging in each level; ii) use fast parallel superpixel generation methods [1, 34] to group image pixels to initial regions before further merging.

The pipeline of our method is shown in Fig. 2, with example results displayed in Fig. 1 and the algorithm listed in Algorithm 1. In the first step, the GPU-SLIC method [1, 34] is exploited to over-segment an input image into superpixels, which serve as seed regions in the 1^{st} level $\mathcal{S}_1 = \{R_1^{(1)}, R_2^{(1)}, \ldots, R_{K_l}^{(1)}\}$.

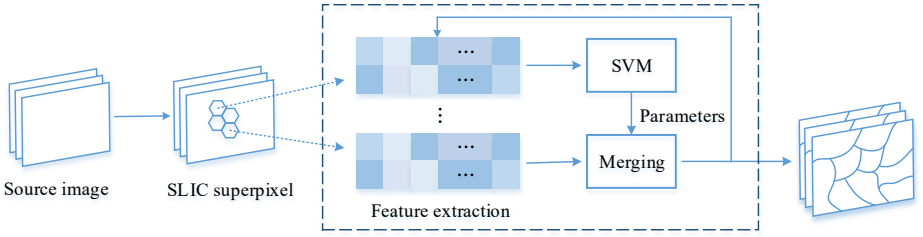


Fig. 2: Pipeline of our methods.

In the subsequent steps, both internal and marginal features (see also Sec. 3.4) are extracted. Using support vector machine (SVM) regressors, we learn from training data (see also Sec. 3.3) how to map feature vectors $\boldsymbol{T}_{i,j}^{(l)}$ to suitable distance measure between regions $R_i^{(l)}$ and $R_j^{(l)}$. We progressively merge regions in $\mathcal{S}_l$ to arrive at a coarser segmentation $\mathcal{S}_{l+1}$, following the efficient graph based (EGB) image segmentation framework [17] with the graph defined in Equ. (2).

Our design principle is motivated by a recent trend of using discriminative learning approach to find proper feature combination for various vision tasks [4,22,30]. A number of psychophysics studies [36] suggest that humans use multi-cues to separate objects in natural scenes. Compared with an ad-hoc design, extracting image features and allowing the data to speak for themselves is proven to be an appropriate way of learning how to combine different visual cues [4,30]. Our system design is also motivated by an observation that image features play different roles at different scales, when deciding whether two regions should be merged. At a fine scale, *e.g.* pixel level, color similarity and spatial distance are important, which is observed in many state-of-the-art image segmentation methods [1, 17]. With region merging/grouping progressing to a coarser level, texture similarity, edges between regions and other cues become more important deciding factors to judge whether two regions should be merged.

Instead of learning a single rule for cue/feature combination across all levels, [4,30], we experiment an alternative approach in which iteratively updating region features and their combination weights is implemented (see also Fig. 3).

Algorithm 1 HFS for region merging

Input: image I, weights $\mathbf{w}$, iteration L
Output: segmentation $\mathcal{S}_L$
Initialization: $\mathcal{S}_l = \{R_1^{(l)}, R_2^{(l)}, \ldots, R_{K_l}^{(l)}\} \Leftarrow$ GPU-SLIC(I) [1,34]
for $l = \{1, 2, \ldots, L-1\}$ **do**
  for each $(R_i^{(l)}, R_j^{(l)}) \in \mathcal{A}_l$ **do**
    $s_{i,j}^{(l)} \Leftarrow (\boldsymbol{T}_{i,j}^{(l)})^T \cdot \mathbf{w}^{(l)}$, see also Sec. 3.4 & 3.3 for $\boldsymbol{T}_{i,j}^{(l)}$ and $\mathbf{w}^{(l)}$ respectively
  end for
  $\mathcal{S}_{l+1} = \text{EGB}(\mathcal{A}_l, s_{i,j}^{(l)})$ [17]
end for

We see favorable aspect of our approach in the experiments. Based on this point, we design a hierarchical architecture in which multiple levels are engaged, followed by recursive region merging [17] and feature updating.

One thing worth noting is that we only extract features that are simplistic and suitable for parallel computing on modern GPUs. More informative CNN features can be engaged in an e.g. end-to-end edge detection system HED [46], to improve the segmentation results.

Our experimental results indicate that the F-measure [4]

$$F_\beta = \frac{2 \cdot Precision \times Recall}{Precision + Recall}, \tag{3}$$

can be increased by 6% when HED is used. However, because of the overhead of HED ($0.4s$ per-image), as opposed to $0.02s$ in our vanilla version, we make using HED an optional choice.

3.3 Parameter Learning

As described above, given a set of initial regions, we learn an edge weight $\mathbf{w}_{i,j}^{(l)} \in \mathbf{w}^{(l)}$ between every region pair $(R_i^{(l)}, R_j^{(l)}) \in \mathcal{A}_l$. Since every region pair is associated with a feature vector $\boldsymbol{T}_{i,j}^{(l)}$, our next step is to provide a label for each region pair at level l. Since the initial regions of each level may have irregular shapes, we use the F-measure to help determine the ground truth label of each region pair in $\mathcal{A}_l$.

We first calculate the F-measure of the initial segmentation at level l, denoted as $F_{init}^{(l)}$. Then, for each region pair $(R_i^{(l)}, R_j^{(l)})$ in $\mathcal{A}_l$, we compute the F-measure repeatedly after merging $(R_i^{(l)}, R_j^{(l)})$. If the F-measure after merging $(R_i^{(l)}, R_j^{(l)})$ is greater than $F_{init}^{(l)}$, the corresponding label $y_{i,j}^{(l)}$ of $(R_i^{(l)}, R_j^{(l)})$ will be assigned to 0. Otherwise, $y_{i,j}^{(l)}$ will be assigned to 1. We adopt support vector machine (SVM) regressor to learn feature weights $\mathbf{w}^{(l)}$.

3.4 Feature Extraction

Our system explores a group of simple features that can be efficiently calculated on modern GPUs. Both internal and marginal features are considered here. Tab. 1 lists the features we have considered. We discuss below the details of these features used in our system.

Brightness and Colors The brightness and color cues in the CIELAB color space have been proved to be very useful [4,30]. We use mean L*a*b* values to represent the color of a segment. In order to tolerate variations in the relative weight of brightness and colors, we use both the Euler distance (d_c) and the distances in each channel (d_l, d_a, d_b) for two adjacent segments.

Average gradient maximum along boundary Previous works have shown that gradient information is an important cue in boundary detection. Instead of using gradients directly, we use gradients after non-maxima suppression. For

Table 1: Features for adjacent regions

Feature names	Dimension	Notation
differences in each channel of CIELAB	3	d_l, d_a, d_b
Euler distance of CIELAB values	1	d_c
average gradient maximum along boundary	1	d_g
χ^2 distance between RGB histograms	1	χ^2_h
χ^2 distance between gradient histograms	1	χ^2_H
variances of RGB values	3	s_r, s_g, s_b
variances of CIELAB values	3	s'_l, s'_a, s'_b
average HED maximum along boundary	1	d_h

adjacent segments R_i and R_j, the computation starts by placing a small circular disc at the pixel $p_k \in \Gamma$, where Γ represents their boundaries. Then we calculate the maximum gradient $\delta'(p_k)$ in the disc. Thereby, the average $\delta'(p_k)$ is computed as the gradient difference $d_g(R_i, R_j)$.

χ^2 distance between RGB histograms To make use of the details of color information, we employ the color histogram that has $8 \times 8 \times 8$ dimensions in the RGB color space. For histograms belonging to adjacent segments, we use χ^2 distance to measure their difference.

χ^2 distance between gradient histograms The χ^2 distance of two segments when computing histograms of oriented gradient for each segment is also an attractive choice.

Variances Variance is a good measure for the fluctuation of a piece of data. We apply variances in the RGB (s_r, s_g, s_b) and CIELAB (s'_l, s'_a, s'_b) color spaces to $R_i \bigcup R_j$, where R_i and R_j are adjacent segments. The magnitude of the variances reflects the similarity between the two segments.

Average HED maximum along boundary The HED feature is computed similarly to gradient feature above. However, because of the extra overhead of HED, we make this choice optional.

The above features play different roles in different levels. The weight comparison of features in the first and second level are shown in Fig. 3 (except for the HED features). To take computational complexity into account, we only choose a small set of features that are easy to calculate instead of using all of them. The top five features are d_l, d_a, d_b, d_c, and d_g. All the experimental results reported in this paper are based on these features.

3.5 Implementation details

To design a practical system, we choose $L = 3$ as a default value in this paper. We do all experiments using a machine with an Intel Xeon CPU E5-2676 v3 @ 2.40GHz and an NVIDIA GeForce GTX 980 Ti. All the running time is reported without data parallelism, except for the objectness proposal application part (Sec. 4.2) which is designed to adherence to the general practice.

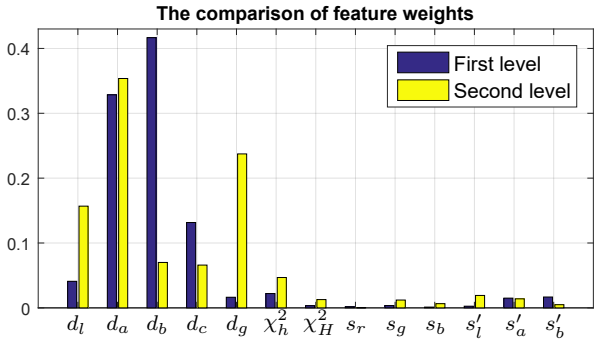


Fig. 3: The weight comparison of the features learned in the 1^{st} and 2^{nd} level.

4 Experiments

4.1 Evaluation

In this part, we evaluate our method on the BSDS500 [4] benchmark, which is widely used to evaluate segmentation and grouping methods. There are two choices of measures: *optimal dataset scale (ODS)* which is the optimization for the entire training dataset, and *optimal image scale (OIS)* which is the optimization for each test image. For boundary assessment, we use the F-measure of precision and recall on the ODS. The region-based measures contain:

- Variation of Information (VI), measuring the distance between *ground truth (GT)* and the proposed segmentation;
- Probabilistic Rand Index (PRI), measuring the pairwise compatibility of element assignment between GT and the proposed segmentation;
- Segmentation Covering (Covering), measuring the average overlap between GT and the proposed segmentation.

See [4] for more details. Fig. 3 shows the weight comparison of the selected features. We can see clearly that the weight importance is different in different levels. To make our results more convincing, we compare our method with approaches [1, 4, 13, 14, 17, 31, 40], the MCG as well as SCG approaches in [5]. All the experiments are accomplished using publicly available source code.

We show the evaluation results on boundary benchmark in Fig. 4, in which all of the execution time is tested without data parallel. We can see that [5] achieves the best result comparing to others. However, its simplified version SCG still needs about 2 seconds to process an image. For this reason, it cannot be employed in nowadays applications in spite of its stroke of genius. Similarly, the accuracy of [4] is very competitive compared with other methods. However, the speed of this method is extremely slow taking about 86 seconds per image. Our approach is hundreds of times faster than [4, 5], achieving 50 fps. When the data parallelism is enabled, the speed can be up to 200+ fps. In addition, our approach can be easily used in almost all the applications nowadays including

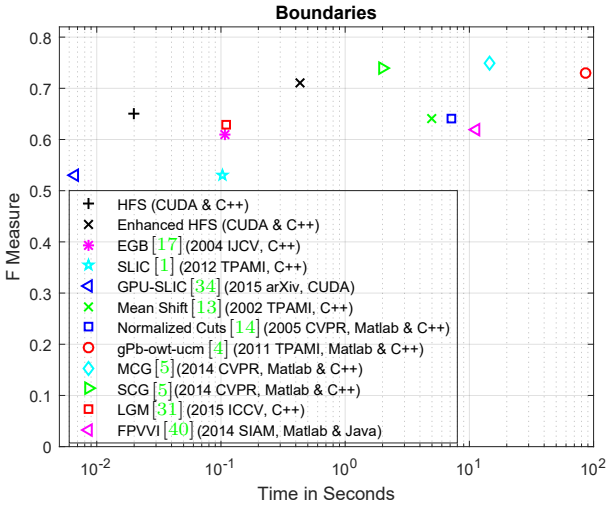


Fig. 4: Experimental evaluation for boundaries on BSDS500 [4] test set. The F-measure is computed by precision and recall at the Optimal Dataset Scale (ODS). And the execution time is tested without data parallel.

some real-time systems. Comparing with some superpixel extraction methods, e.g. [1, 17, 31], Fig. 4 demonstrates that our method is much faster, and more importantly, the accuracy also has a significant improvement. When comparing with the rest three methods [13, 14, 40], our speed advantage is obvious, though the F-measure is only a little higher than them. Others, the F-measure of our enhanced version is very close to the best performance, with very fast speed.

Table 2: Region Benchmarks on the BSDS500 [4]

Methods	GPU	Time(s)	Covering			PRI		VI	
			ODS	OIS	Best	ODS	OIS	ODS	OIS
Human	–	–	0.73	0.73	–	0.87	0.87	1.16	1.16
HFS	Y	0.02	0.56	0.61	0.70	0.81	0.84	1.87	1.68
Enhanced HFS	Y	0.43	0.58	0.65	0.72	0.82	0.86	1.80	1.64
EGB [17]	N	0.11	0.52	0.57	0.69	0.80	0.82	2.21	1.87
SLIC [1]	N	0.10	0.37	0.38	0.48	0.74	0.75	2.56	2.50
GPU-SLIC [34]	Y	0.007	0.34	0.37	0.47	0.73	0.75	2.95	2.81
Mean Shift [13]	N	4.95	0.54	0.58	0.66	0.79	0.81	1.85	1.64
Normalized Cuts [14]	N	7.15	0.45	0.53	0.67	0.78	0.80	2.23	1.89
gPb-owt-ucm [4]	N	86.4	0.59	0.65	0.74	0.83	0.86	1.69	1.48
MCG [5]	N	14.5	0.61	0.66	0.76	0.83	0.86	1.57	1.40
SCG [5]	N	1.98	0.60	0.65	0.74	0.83	0.86	1.63	1.43
LGM [31]	N	0.11	0.52	0.56	0.63	0.78	0.81	1.93	1.79
FPVVI [40]	N	11.3	0.47	0.53	0.62	0.77	0.80	2.10	1.92

Fig. 5: Some examples of EGB, SLIC and our method. The reason why we only compare with these two algorithms is that they are the only two that is effcient enough to be used in applications. Left: Image. Middle left: SLIC. Middle right: EGB. Right: Ours. The regions are represented by their mean color. And all images are from the test set of BSDS500 [4].

Tab. 2 presents region benchmarks on the BSDS500 [4]. From Tab. 2, MCG performs best on all the metrics, but it needs about 15 seconds per image. SLIC

achieves the worst results, although its GPU version can be very fast. It is not difficult to find that our approach is close to the best performance on all these criteria, especially the enhanced version. Thus, we can draw the conclusion that our approach can achieve better trade-off than others in both efficiency and quality. Fig. 5 shows some examples of our method comparing with the other two fast algorithm [1, 17].

The reason for not obtaining the best results on each criterion is two-fold. First, the initial superpixels produced by SLIC are not so desirable. For instance, when the step S is set to 8 pixels, intuitively 2200 superpixels would be produced for each image. However, the boundary recall of SLIC which measures the fraction of the ground truth contours that fall into the eight neighbourhoods of a superpixel boudary, is only 73%. This fact may significantly affect the first-level results of our merging strategy. Second, since [17] is unable to control the compactness of generated superpixels, our merging strategy cannot get the desired regions. More specifically, in [17], only a constant parameter is used to prevent each region from being too large. In fact, this criterion is sometimes not reasonable because of the diversity of input pictures.

Nevertheless, because of our powerful architecture, our results still outperform most existing segmentation approaches. The following parts describe the applications of our approach in both saliency detection and objectness proposal, from which one can find the practicality of our approach.

4.2 Objectness

Generic object proposal generation has been a hot topic in recent years. As a preprocessing step in many applications such as object recognition and detection, it generates a number of bounding boxes that may contain objects. This type of algorithms have been used in many existing object detection methods [43, 45]. It has been shown that these object detection methods can perform better than the classical sliding-window-based paradigm [15, 16, 18].

As for metrics that measure the objectness approaches, we adopt mean average best overlap (MABO) across all the classes [43] and computational efficiency. Cheng *et al.* [12] recently propose a very fast method (BING), which generates box proposals at 300 fps, but this method cannot perform well on MABO benchmark. Chen *et al.* [8] propose a postprocessing approach (MTSE) to refine bounding boxes produced by objectness methods. In their algorithm, they use [17] to generate regions. In order to show the advantages of our segmentation method, we choose [8] as the postprocessing step of [12] and replace [17] with our method.

We extensively evaluate the new system on the challenging PASCAL VOC 2007 dataset [16]. To demonstrate the advantages of our system, we compare our results with some currently influential methods, including [2, 12, 43, 50]. From Fig. 6, one can find that our modified version performs better than the original one [8]. With our segmentation method, the speed has been significantly boosted. We can get competitive boxes at over 100fps, comparing to 0.25s per image as reported in [8]. And not only that, Fig. 6 indicates that the new system using

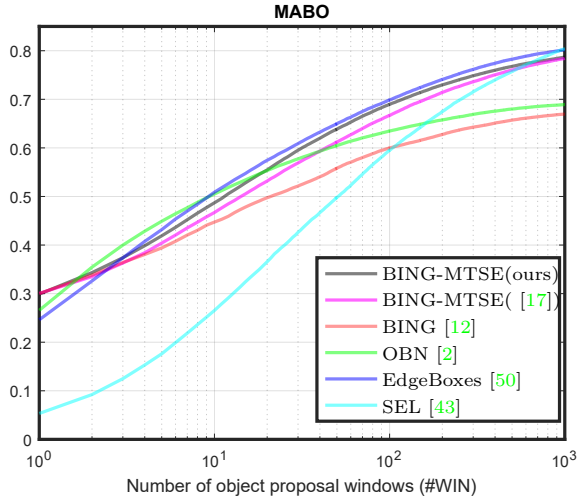


Fig. 6: Tradeoff between MABO and number of proposals using different methods on VOC2007 test set.

our segmentation is one of the best objectness methods in terms of quality. As a result, our new system without doubt can make the best trade-off between efficiency and quality.

4.3 Saliency

In this part, we report the superiority of our method when it is used in another domain of computer vision. Visual saliency has been a fundamental problem in neuroscience, psychology, neural systems, and computer vision for a long time. In computer vision, detecting and segmenting salient objects in natural scenes, also known as salient object detection, has attracted a lot of focused research and has resulted in many applications. However, because most saliency detection methods are region-based, these exist two things as the bottleneck of salient object detection for a long period. First one is the segmentation quality [33] and the other is the computation efficiency. Recently, Jiang *et al.* [22] proposed a supervised learning method (DRFI) to predict a salient score of the regions produced by the popular segmentation method [17], which receives good performance on several popular datasets, such as MSRA10K [10] and PASCAL-S [27]. Here, we replace [17] with our segmentation method as a single level.

For a faithful comparison, we evaluate current popular detection methods [10,11,19,22,29,47,48] on several datasets mentioned above using mean absolute error(MAE) [32], which is introduced to reflect the negative saliency assignments. It is defined between a saliency map S and the binary groundtruth GT as:

$$MAE = \frac{1}{|I|}\sum_{x}|S(I_x) - GT(I_x)|, \tag{4}$$

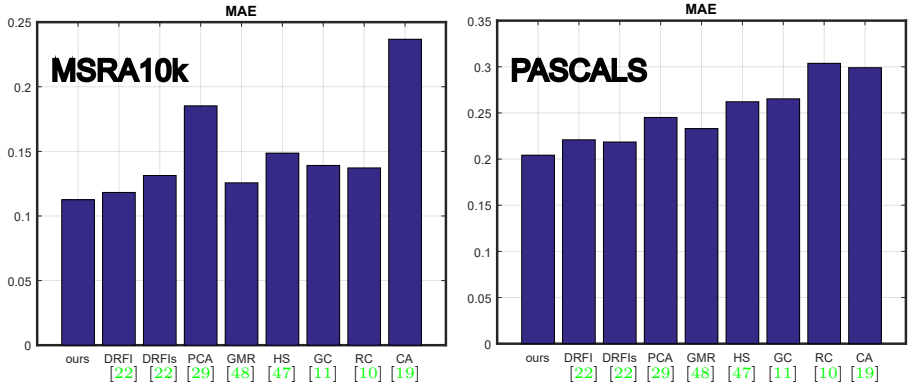


Fig. 7: Mean absolute errors of the state-of-the-art methods on MSRA10K [10] and PASCAL-S [27]. DRFIs is the single level version of DRFI, and note that our method is a single level. The proposed approach consistently achieves the lowest error rates on all datasets.

where $|I|$ is the total number of pixels. The MAE results on these two datasets are shown in Fig. 7. Our method achieves the lowest MAE values on all datasets. Specifically speaking, it decreases by 0.57% and 1.43% over the second best algorithms in terms of MAE scores. This means that its predicted saliency pixels are closest to the ground truth.

5 Discussion

In this paper, we have proposed a hierarchical method for image segmentation. We design a hierarchical architecture to enjoy the benefits of engaging different feature setting in different scale levels. In addition, we explore the capability of modern GPUs to efficiently compute a set of simple but useful features. Our approach produces high quality hierarchical regions with substantial speed-up when compared with previous state-of-the-art works. Evaluation results on standard benchmark (BSDS500 [4]) show that our method achieves a favorable trade-off between efficiency and quality. When plugged into other computer vision tasks such as objectness and saliency detection, our method improves their performance. To encourage future works, we make the source code of this work publicly available at `http://mmcheng.net/hfs/`.

Acknowledgments

We would like to thank the anonymous reviewers for their useful feedbacks. This research was sponsored by NSFC (NO. 61572264), Huawei Innovation Research Program (HIRP), and CAST young talents plan.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. IEEE TPAMI 34(11), 2274–2282 (2012)
2. Alexe, B., Deselaers, T., Ferrari, V.: Measuring the objectness of image windows. IEEE 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. IEEE TPAMI 34(2), 315–327 (2012)
4. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE TPAMI 33(5), 898–916 (2011)
5. Arbelaez, P., Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: IEEE CVPR. pp. 328–335 (2014)
6. Chang, J., Wei, D., Fisher, J.W.: A video representation using temporal superpixels. In: IEEE CVPR. pp. 2051–2058 (2013)
7. Chen, T., Cheng, M.M., Tan, P., Shamir, A., Hu, S.M.: Sketch2photo: internet image montage. ACM TOG 28(5), 124 (2009)
8. Chen, X., Ma, H., Wang, X., Zhao, Z.: Improving object proposals with multithresholding straddling expansion. In: IEEE CVPR (2015)
9. Cheng, J., Liu, J., Xu, Y., Yin, F., Wong, D.W.K., Tan, N.M., Tao, D., Cheng, C.Y., Aung, T., Wong, T.Y.: Superpixel classification based optic disc and optic cup segmentation for glaucoma screening. IEEE Transactions on Medical Imaging 32(6), 1019–1032 (2013)
10. Cheng, M.M., Mitra, N.J., Huang, X., Torr, P.H., Hu, S.: Global contrast based salient region detection. IEEE TPAMI 37(3), 569–582 (2015)
11. Cheng, M.M., Warrell, J., Lin, W.Y., Zheng, S., Vineet, V., Crook, N.: Efficient salient region detection with soft image abstraction. In: IEEE ICCV. pp. 1529–1536 (2013)
12. Cheng, M.M., Zhang, Z., Lin, W.Y., Torr, P.: Bing: Binarized normed gradients for objectness estimation at 300fps. In: IEEE CVPR. pp. 3286–3293 (2014)
13. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE TPAMI 24(5), 603–619 (2002)
14. Cour, T., Benezit, F., Shi, J.: Spectral segmentation with multiscale graph decomposition. In: IEEE CVPR. vol. 2, pp. 1124–1131 (2005)
15. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE CVPR. pp. 248–255 (2009)
16. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IEEE ICCV 88(2), 303–338 (2010)
17. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV 59(2), 167–181 (2004)
18. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE CVPR. pp. 580–587 (2014)
19. Goferman, S., Zelnik-Manor, L., Tal, A.: Context-aware saliency detection. IEEE TPAMI 34(10), 1915–1926 (2012)
20. Hoiem, D., Efros, A., Hebert, M., et al.: Geometric context from a single image. In: IEEE ICCV. vol. 1, pp. 654–661 (2005)
21. Hu, S.M., Zhang, F.L., Wang, M., Martin, R.R., Wang, J.: Patchnet: a patch-based image representation for interactive library-driven image editing. ACM TOG 32(6), 196 (2013)

22. Jiang, H., Wang, J., Yuan, Z., Wu, Y., Zheng, N., Li, S.: Salient object detection: A discriminative regional feature integration approach. In: IEEE CVPR. pp. 2083–2090 (2013)
23. Jiang, Z., Davis, L.S.: Submodular salient region detection. In: IEEE CVPR. pp. 2043–2050 (2013)
24. Juneja, M., Vedaldi, A., Jawahar, C., Zisserman, A.: Blocks that shout: Distinctive parts for scene classification. In: IEEE CVPR. pp. 923–930 (2013)
25. Kohli, P., Torr, P.H., et al.: Robust higher order potentials for enforcing label consistency. IJCV 82(3), 302–324 (2009)
26. Li, K., Zhu, Y., Yang, J., Jiang, J.: Video super-resolution using an adaptive superpixel-guided auto-regressive model. Pattern Recognition 51, 59–71 (2016)
27. Li, Y., Hou, X., Koch, C., Rehg, J., Yuille, A.: The secrets of salient object segmentation. In: IEEE CVPR. pp. 280–287 (2014)
28. Maire, M., Yu, S.X.: Progressive multigrid eigensolvers for multiscale spectral segmentation. In: IEEE ICCV. pp. 2184–2191 (2013)
29. Margolin, R., Tal, A., Zelnik-Manor, L.: What makes a patch distinct? In: IEEE CVPR. pp. 1139–1146 (2013)
30. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE TPAMI 26(5), 530–549 (2004)
31. Nguyen, R.M., Brown, M.S.: Fast and effective l0 gradient minimization by region fusion. In: IEEE ICCV. pp. 208–216 (2015)
32. Perazzi, F., Krähenbühl, P., Pritch, Y., Hornung, A.: Saliency filters: Contrast based filtering for salient region detection. In: IEEE CVPR. pp. 733–740 (2012)
33. Qi, W., Cheng, M.M., Borji, A., Lu, H., Bai, L.F.: Saliencyrank: Two-stage manifold ranking for salient object detection. Computational Visual Media 1(4), 309–320 (2015)
34. Ren, C.Y., Prisacariu, V.A., Reid, I.D.: gslicr: Slic superpixels at over 250hz. arXiv preprint arXiv:1509.04232 (2015)
35. Ren, Z., Shakhnarovich, G.: Image segmentation by cascaded region agglomeration. In: IEEE CVPR. pp. 2011–2018 (2013)
36. Rivest, J., Cabanagh, P.: Localizing contours defined by more than one attribute. Vision research 36(1), 53–66 (1996)
37. Russell, C., Kohli, P., Torr, P.H., et al.: Associative hierarchical crfs for object class image segmentation. In: IEEE ICCV. pp. 739–746 (2009)
38. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE TPAMI 22(8), 888–905 (2000)
39. Song, X., Zhang, J., Han, Y., Jiang, J.: Semi-supervised feature selection via hierarchical regression for web image classification. Multimedia Systems 22(1), 41–49 (2016)
40. Storath, M., Weinmann, A.: Fast partitioning of vector-valued images. SIAM Journal on Imaging Sciences 7(3), 1826–1852 (2014)
41. Sun, J., Ponce, J.: Learning discriminative part detectors for image classification and cosegmentation. In: IEEE ICCV. pp. 3400–3407 (2013)
42. Taylor, C.J.: Towards fast and accurate segmentation. In: IEEE CVPR. pp. 1916–1922 (2013)
43. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. IJCV 104(2), 154–171 (2013)
44. Wang, S., Lu, H., Yang, F., Yang, M.H.: Superpixel tracking. In: IEEE ICCV. pp. 1323–1330 (2011)

45. Wang, X., Yang, M., Zhu, S., Lin, Y.: Regionlets for generic object detection. In: IEEE ICCV. pp. 17–24 (2013)
46. Xie, S., Tu, Z.: Holistically-nested edge detection. In: IEEE ICCV. pp. 1395–1403 (2015)
47. Yan, Q., Xu, L., Shi, J., Jia, J.: Hierarchical saliency detection. In: IEEE CVPR. pp. 1155–1162 (2013)
48. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: IEEE CVPR. pp. 3166–3173 (2013)
49. Zhang, L., Gao, Y., Xia, Y., Lu, K., Shen, J., Ji, R.: Representative discovery of structure cues for weakly-supervised image segmentation. IEEE Transactions on Multimedia 16(2), 470–479 (2014)
50. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: ECCV. pp. 391–405 (2014)