# An End-to-End Tree Based Approach for Instance Segmentation

K. V. Manohar[1](✉) and Yusuke Niitani[2](✉)

[1] IIT Kharagpur, Kharagpur, India
kvmanohar22@gmail.com
[2] Preferred Networks Inc., Tokyo, Japan
niitani@preferred.jp

**Abstract.** This paper presents an approach for bottom-up hierarchical instance segmentation. We propose an end-to-end model to estimate energies of regions in an hierarchical region tree. To this end, we introduce a Convolutional Tree-LSTM module to leverage the tree-structured network topology. For constructing the hierarchical region tree, we utilize the accurate boundaries predicted from a pre-trained convolutional oriented boundary network. We evaluate our model on PASCAL VOC 2012 dataset showing that we obtain good trade-off between segmentation accuracy and time taken to process a single image.

## 1 Introduction

In this work we address the task of *instance segmentation* which involves segmenting each individual instance of a semantic class in an image. Many top-down approaches to this problem are based on object detection pipelines [1,2] and each box is refined to generate a segmentation. Further, these methods do not consider entire image but rather independent proposals and as a result cannot handle occlusions between different objects. Since these methods are based on initial detections, they cannot recover from false detections motivating an approach that reasons globally.

A key aspect of our approach is to leverage the hierarchical segmentation trees [3] to sample potential object instances. To this end, we propose a new bottom-up approach to parse the regions in an hierarchical region tree. At the core of our approach lies Convolutional Tree-LSTM module which estimates the energies of the regions taking into account the entire image and tracking temporal relations across regions through different levels of the tree. Unlike MCG [4], that uses hand engineered features to generate object candidates, we exploit rich features learnt by Convolutional Neural Networks to sample object instances. Further, MCG involves complex pipeline involving proposal generation and ranking. The resulting system is very slow and takes more than 9.9 s for candidate generation

K. V. Manohar—Work done when the author was an intern at Preferred Networks inc., Japan.

alone. Ours on the other hand is trained end-to-end and on average takes 0.06 s at test time.

Our paper is outlined as follows. We begin by reviewing related work in Sect. 2. In Sect. 3 we describe the details of our approach. In Sect. 4, we dwell into implementation details. We investigate the performance of our method both qualitatively and quantitatively in Sect. 5. Finally, we conclude in Sect. 6.

## 2 Related Work

Our work is closely related to bottom-up methods exploiting superpixels [5]. Pham et al. [6] proposed a dynamic programming based approach to image segmentation by constructing a hierarchical segmentation tree. An unified energy function jointly quantifies geometric goodness-of-fit and objectness measure. A top-down traversal through the tree comparing the energies of the current node and its subtree results in optimal tree cut. Kirillov et al. [7] impose graph structure on the superpixels and formulate instance estimation as a MultiCut problem. One of the limitations of this method however is that, it cannot find instances that are formed by disconnected regions in the image. Unlike these methods, by training our model end-to-end we can find such instances as discussed in Sect. 6.

## 3 Method

Given an input image $\mathcal{I}$, our goal is to segment the image into semantically meaningful non-overlapping regions. Figure 1 depicts the overview of our method. Henceforth, we adopt the following notation. For a given $\mathcal{I}$, let $\mathcal{T}$, $L = \{1, 2, \ldots, l_{max}\}$, $\mathcal{R} = \{r_1, r_2, \ldots, r_N\}$, $\mathcal{F} = \{F_{r_1}, F_{r_2}, \ldots, F_{r_N}\}$ and $\mathcal{C} = \{C_{r_1}, C_{r_2}, \ldots, C_{r_N}\}$ represent the hierarchical tree, set of distinct levels, set of regions in the tree, corresponding features for the regions and children of the regions in the tree respectively. For each level $0 < l \leq l_{max}$, we denote the set of regions, corresponding features and the threshold at this level as $\mathcal{R}_l = \{r_1^l, r_2^l, \ldots, r_{N_l}^l\} \subseteq \mathcal{R}$, $\mathcal{F}_l = \{F_{r_1^l}, F_{r_2^l}, \ldots, F_{r_{N_l}^l}\}$ and $\alpha_l$ respectively. Tree cut at a level $l^{'}$ for a horizontal cut-threshold $\lambda_{cut} = \alpha_{l^{'}}$ results in a new set of levels $L^{'} = \{l | l \geq l^{'}\}$.

### 3.1 Feature Extraction

We first extract features $\mathbf{F}$ by passing input image $\mathcal{I}$ through a series of convolutions. For a given region $r \in \mathcal{R}$ in the tree, we generate a tightest bounding box $b_r$ covering the non-linear boundary of $r$. We then extract a fixed spatial dimensional feature map $F_r^*$(e.g., $7 \times 7$) from $\mathbf{F}$ corresponding to $b_r$. Our approach in extracting $F_r^*$ is similar to ROIAlign layer [1]. Additionally, we mask out the features corresponding to the region $b_r \setminus r$ giving rise to the final feature map $F_r$.

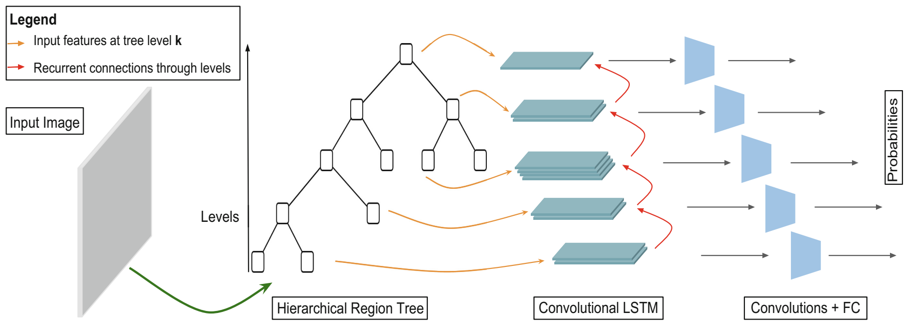

**Fig. 1.** Overview of our method. We (1) construct hierarchical region tree using Ultrametric Contour Map (UCM), (2) estimate energies of each region in the tree starting from level 1 at the bottom and all the way to the top, and (3) threshold the regions based on the energies.

## 3.2 Convolutional Tree-LSTM Module

The motivation behind the method is to estimate how the probabiliy distribution over the categories change when a new region is added to the region under consideration in the subsequent levels. The model implicitly learns the temporal relations which lead to the formation of a given region.

We process the hierarchical tree $\mathcal{T}$ starting from level $l^{'}$ which corresponds to the initial cut-threshold $\lambda_{cut} = \alpha_{l'}$ using Convolutional Tree-LSTM predicting softmax probabilities for each region $r \in \mathcal{R}_l$ at all the levels $l \in L^{'}$ in order. Input to the LSTM at each level $l$ are the features $\mathcal{F}_l$. Equations 1–7 summarizes the forward propagation through the LSTM module. For $j$th region at level $l$,

$$\tilde{h_j^l} = \sum_{k \in C_{r_j^l}} h_k^l, \tag{1}$$

$$i_j^l = \sigma(W^i * F_{r_j^l} + U^i * \tilde{h_j^l} + b^i), \tag{2}$$

$$f_{jk}^l = \sigma(W^f * F_{r_j^l} + U^f * h_k^l + b^f) \quad \forall k \in C_{r_j^l}, \tag{3}$$

$$o_j^l = \sigma(W^o * F_{r_j^l} + U^o * \tilde{h_j^l} + b^o), \tag{4}$$

$$u_j^l = \tanh(W^u * F_{r_j^l} + U^u * \tilde{h_j^l} + b^u), \tag{5}$$

$$c_j^l = i_j^l \odot u_j^l + \sum_{k \in C_{r_j^l}} f_{jk}^l \odot c_k^l, \tag{6}$$

$$h_j^l = o_j^l \odot \tanh(c_j^l), \tag{7}$$

where $*, \odot$ denote convolution operation and Hadamard product respectively. We do the above for each region $j$ and $\forall\ l \in L^{'}$. For a region $j$ at level $l$, $c_k^l, h_k^l \forall k \in C_{r_j^l}$ are initialized to zeros provided they are the leaves of the tree and for the rest of the regions, $c_k^l, h_k^l$ are governed by the Eqs. 6 and 7 respectively.

Figure 2 depicts analysis on variation of sequence length and number of regions considered for different horizontal cuts.

On top of the LSTM module, we apply series of convolutions and fully connected layers which take input as $h_j^l$ and predict probabilities.

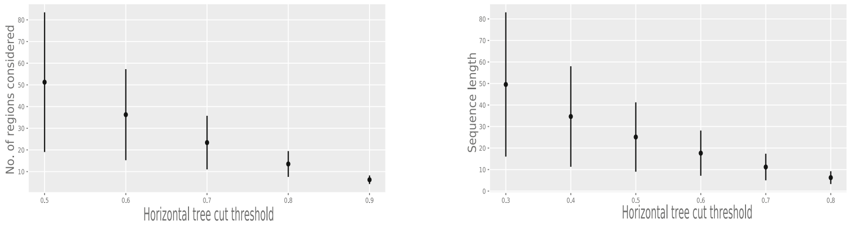

**Fig. 2.** Variation of number of regions considered and sequence length for different initial horizontal cut thresholds.

### 3.3 Objective Formulation

For a given image $\mathcal{I}$, let $\mathcal{M} = \{m_1, m_2, \dots m_M\}, L^G = \{l_1, l_2, \dots l_M\}$ be the set of ground truth masks and one-hot labels respectively. For each mask $m_i$, we construct the positive set $\mathcal{P}_i^+ = \{p_1^i, p_2^i, \dots p_{N_i}^i\}$ which consists of probabilities of regions from $\mathcal{R}$ whose IoU with $m_i$ is greater than $\lambda_+$. Similarly, we construct $\mathcal{P}^- = \{p_1^-, p_2^-, \dots p_{N_-}^-\}$ consisting of probabilities of regions from $\mathcal{R}$ whose IoU with all $m_i$ is less than $\lambda_-$. We then formulate the loss as follows,

$$\mathcal{L} = -\frac{1}{M} \sum_{i=1}^{M} \sum_{r=1}^{|\mathcal{P}_i^+|} l_i^T \log(p_r^i) - \lambda \sum_{r=1}^{|\mathcal{P}^-|} \sum_{c=1}^{C} I_c^b \log(p_r^-), \tag{8}$$

where $I_c^b$ is 1 if class $c$ corresponds to the background label $b$ and $T$ represents the transpose of vector. The hyperparameter $\lambda$ in Eq. 8 controls the balance between positive and negative regions.

## 4 Implementation Details

### 4.1 Network Architecture

We use the pre-trained COB network for estimating contours which is a ResNet50 model. Features $\mathbf{F}$ are extracted from *res3* layer of ResNet50 model having spatial resolution of $28 \times 28$. ROIAlign extracts features having a fixed spatial resolution of $7 \times 7$. All the convolutions within the LSTM have kernel size of $3 \times 3$, stride 1 and use zero-padding. On top of convolutional LSTM, we have 2 $3 \times 3$ convolutions and 2 fully connected layers predicting softmax probabilities.

## 4.2 Training Details

We set the parameters $\lambda_+$, $\lambda_-$, $\lambda$ to 0.7, 0.3 and 0.2 respectively in all our experiments. We train Convolutional LSTM and subsequent layers from scratch with a batch size of 1, initial learning rate of 0.001 and decay it by a factor of 0.1 after every 20 epochs. We experiment over various initial cut-thresholds from $\lambda_{cut} = 0.3$ to $\lambda_{cut} = 0.9$ in steps of 0.1.

## 5 Experiments

We use the pretrained COB network to predict the contours which was trained on PASCAL Context dataset. We train our Convolutional Tree-LSTM and subsequent layers on PASCAL VOC 2012 dataset. We evaluate our model on PASCAL VOC 2012 val dataset using average precision, Jaccard Index and time taken to process an image as evaluation metrics. Table 1 compares the time taken to process a single image by different methods. Figure 3 denotes the precision-recall curves for all the classes.

On the VOC 2012 val set, our best performing model scores 48% mAP. Our model struggles on categories like *bicycle*, *chair*. However on categories like *train* and *plane*, our model achieves higher performance. Table 2 summarizes the average precision for all the categories. We further compare Jaccard Index with MCG and is presented in Table 3 (Fig. 4).

**Table 1.** Time taken to process a single image in seconds.

| Method | Hierarchical segmentation[a] | Candidate generation[b] | Total |
|---|---|---|---|
| MCG [4] | $24.4 \pm 3.6$ | $9.9 \pm 3.5$ | $34.3 \pm 6.2$ |
| SCG [4] | $3.2 \pm 0.4$ | $1.5 \pm 0.5$ | $4.7 \pm 0.7$ |
| Scene-cut [6] | 0.79 | 3.76 | 4.55 |
| **Ours** | 0.79 | 0.06 | 0.85 |

[a] Involves estimating contours, UCM and constructing hierarchical region tree
[b] Involves estimating energies of regions and optimal tree cut

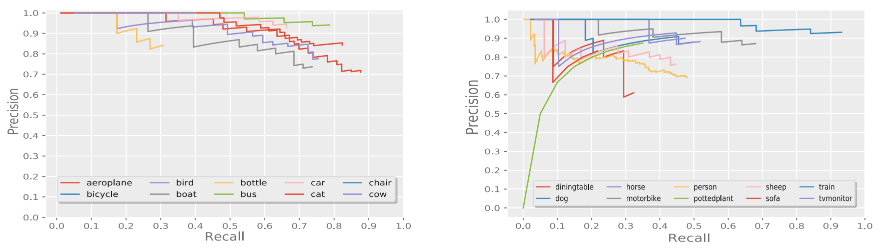

**Fig. 3.** Precision-recall curves for all categories in VOC 2012 val dataset.

**Table 2.** Variation of average precision for different tree cut thresholds.

| Cut threshold | Plane | Bicycle | Bird | Boat | Bottle | Bus | Car | Cat | Chair | Cow | Table | Dog | Horse | MBike | Person | Plant | Sheep | Sofa | Train | TV | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.5 | 0.72 | 0 | 0.5 | 0.44 | 0.16 | 0.73 | 0.46 | 0.63 | 0.01 | 0.26 | 0.12 | 0.37 | 0.32 | 0.53 | 0.29 | 0.15 | 0.39 | 0.15 | 0.71 | 0.3 | 0.36 |
| 0.6 | 0.72 | 0 | 0.49 | 0.45 | 0.15 | 0.74 | 0.44 | 0.62 | 0.01 | 0.27 | 0.12 | 0.37 | 0.29 | 0.55 | 0.29 | 0.16 | 0.39 | 0.15 | 0.69 | 0.31 | 0.36 |
| 0.7 | 0.79 | 0 | 0.75 | 0.54 | 0.14 | 0.68 | 0.63 | 0.76 | 0 | 0.5 | 0.24 | 0.61 | 0.47 | 0.74 | 0.36 | 0.28 | 0.49 | 0.17 | 0.93 | 0.39 | **0.47** |
| 0.8 | 0.81 | 0 | 0.7 | 0.67 | 0.29 | 0.78 | 0.65 | 0.79 | 0 | 0.4 | 0.29 | 0.47 | 0.46 | 0.65 | 0.38 | 0.31 | 0.39 | 0.2 | 0.92 | 0.48 | **0.48** |
| 0.9 | 0.68 | 0 | 0.47 | 0.38 | 0.04 | 0.5 | 0.39 | 0.49 | 0 | 0.22 | 0.03 | 0.19 | 0.18 | 0.42 | 0.29 | 0.1 | 0.44 | 0.11 | 0.66 | 0.23 | 0.3 |

**Table 3.** Comparison of Jaccard Index for varying number of regions considered from the tree. (*N*, *std* stand for number of regions consdidered and standard deviation respectively)

| Method | N | std | Plane | Bicycle | Bird | Boat | Bottle | Bus | Car | Cat | Chair | Cow | Table | Dog | Horse | MBike | Person | Plant | Sheep | Sofa | Train | TV | Global |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MCG [4] | 100 | 0 | 70.2 | 38.8 | 73.6 | 67.7 | 55.3 | 68.5 | 50.6 | 82.4 | 54.4 | 78.1 | 67.7 | 77.7 | 69.3 | 66.3 | 59.9 | 51.4 | 70.2 | 74.1 | 72.6 | 78.1 | 63.7 |
| Ours | 51 | 32 | 68 | 15.2 | 64.7 | 58 | 26.3 | 73.3 | 50.9 | 73.2 | 11 .1 | 41.5 | 26.2 | 58.4 | 52.1 | 55.9 | 40.9 | 29.1 | 52.3 | 32.6 | 78.4 | 47.4 | 47.8 |
|  | 36 | 21 | 68.5 | 16.3 | 63.9 | 56.1 | 24.5 | 72.6 | 50.5 | 73.8 | 11 | 42 | 27.6 | 58.9 | 49.9 | 57.4 | 41.2 | 31.8 | 51.9 | 32.7 | 77.3 | 49.3 | 47.9 |
|  | 23 | 12 | 69.6 | 14.9 | 67.6 | 57.3 | 34.5 | 72 | 57.2 | 77.3 | 10.2 | 48.8 | 29.2 | 64.8 | 57.4 | 58.6 | 43.8 | 31.2 | 55.4 | 38.9 | 76.2 | 48.9 | **50.7** |
|  | 14 | 6 | 68.2 | 15.5 | 65.4 | 56.3 | 30.8 | 73 | 53.8 | 77 | 11.7 | 46.8 | 34.3 | 67 | 57.4 | 58.6 | 44 | 30.8 | 56 | 41.6 | 73.1 | 51.2 | **50.6** |

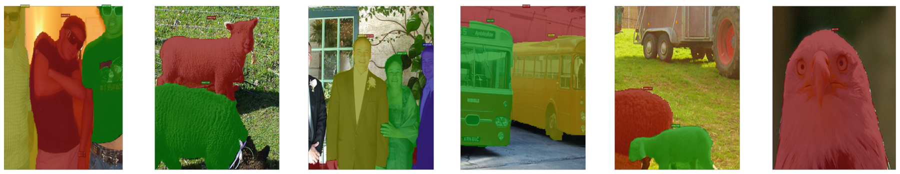
**Fig. 4.** Qualitative results on VOC 2012 val set

## 6 Conclusions

We proposed an unique approach for bottom-up instance segmentation which overcomes the limitations of the current bottom-up and top-down approaches. Our method produces comparative results with good trade-off between segmentation accuracy and processing time. We would like to further investigate an end-to-end network predicting contours in tandem with the estimation of energies of regions. This leads to prediction of semantically accurate contours resulting in high-quality hierarchical region tree further aiding the estimation of energies.

## References

1. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2980–2988, October 2017
2. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4438–4446 (2017)
3. Arbelaez, P., Maire, M., Fowlkes, C.C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
4. Arbeláez, P., Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: Computer Vision and Pattern Recognition (2014)
5. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vis. **59**(2), 167–181 (2004)

6. Pham, T., Do, T.T., Sünderhauf, N., Reid, I.: SceneCut: joint geometric and object segmentation for indoor scenes. In: 2018 IEEE International Conference on Robotics and Automation (ICRA) (2018)
7. Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B., Rother, C.: InstanceCut: from edges to instances with multicut. In: CVPR (2017)