Semantic Segmentation With Object Clique Potentials

Xiaojuan Qi Jianping Shi Shu Liu Renjie Liao Jiaya Jia
The Chinese University of Hong Kong
{xjqi, jpshi, sliu, rjliao, leojia}@cse.cuhk.edu.hk

Abstract

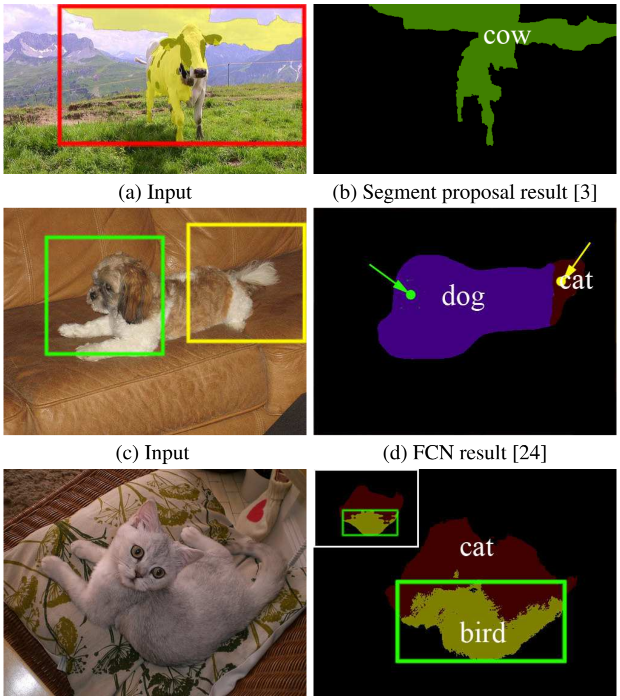
We propose an object clique potential for semantic segmentation. Our object clique potential addresses the misclassified object-part issues arising in solutions based on fully-convolutional networks. Our object clique set, compared to that yielded from segment-proposal-based approaches, is with a significantly smaller size, making our method consume notably less computation. Regarding system design and model formation, our object clique potential can be regarded as a functional complement to local-appearance-based CRF models and works in synergy with these effective approaches for further performance improvement. Extensive experiments verify our method.

1. Introduction

Semantic segmentation is a fundamental task in computer vision that involves labeling each pixel in an image to a category. It relates to the tasks of segmentation, image classification and object detection, but differs from them by nature. Semantic segmentation predicts the additional category information that is not involved in bottom-up segmentation, and classifies multiple objects together with their locations. This is also different from image classification. Compared to object detection, it produces more accurate pixel-level location, and contains additional background class.

In recent years, deep convolutional neural networks (CNN) [17, 32] quicken the development of semantic segmentation systems [24, 3]. One stream is to directly adopt CNN to classify segment proposals generated by objectiveness approaches [7, 11, 3]. These methods enjoy the advantage to classify complete and tight object segment proposals. But there are still two main issues that possibly influence the performance.

First, the computation cost is relatively heavy. Even at test time, around 2,000 segment proposals need to be evaluated in the deep neural networks. Reducing the number of segment proposals could decrease the recall. Second, the initial bottom-up segmentation results almost determine the final structure. When the initial segments go wrong, such as that shown in Fig. 1(a) and (b), the system cannot correct the problematic proposals. Such cases are common in complex-scene images.



(a) Input (b) Segment proposal result [3]
(c) Input (d) FCN result [24]
(e) Input (f) CRF after FCN result [2]

Figure 1. (a) and (b) illustrate problematic segment proposals due to inappropriate initial segments. (c) and (d) show an example difficult for the FCN system. The receptive fields of predicted point in green and yellow in (d) correspond to the bounding box in (c). (e) and (f) demonstrate that the CRF model cannot correct large mistakes – top-left of (f) is the initial FCN prediction result.

The other line [24] is to replace the fully connected layers with convolution ones to produce a fully convolutional network (FCN) for dense prediction. It solves the efficiency problem by reusing the convolution output. The end-to-end dense prediction also leads to flexible segment prediction. But when two categories have similar parts, such as dog and cat, the sliding window receptive field could be misled for part identification due to unawareness of global clues.

Fig. 1(c) and (d) contain a failure case. The tail of the dog is recognized as a cat. Our experiments show that about 65.5% of the failure images in the FCN system are caused by this problem.

Incorporating the local appearance relationship via a CRF model [2] can possibly correct erroneous boundary labels. It is locally effective because if the majority of an object is misclassified, the errors are hardly corrected. Fig. 1(e) and (f) demonstrate this finding, where the initial result in top-left region of (f) misclassifies the bottom of the cat. The CRF model cannot handle it since the mistake is no longer local.

To build an efficient system with flexible segmentation and global object-level clues, we propose a framework with object regularization. It inherits efficiency and flexible segmentation of a FCN-type model, and incorporates object-level information for global regularization. Our experiments show this new object-level regularization covers 84.7% of the overall objects in VOC segmentation dataset. Therefore, most FCN failure cases have the chance to be corrected by this framework. Moreover, our approach does not conflict with the CRF model. Thus its local correction ability can be similarly introduced in our framework.

Our contributions are as follows. First, we propose the object regularization for FCN semantic segmentation. Second, we include an efficient optimization procedure to update object-level and CRF-based regularization iteratively. Finally, we conduct experiments on many data and find our method suitably deals with several previous failure cases.

2. Related Work

Segmenting images with semantic labels is one of the ultimate goals towards image understanding. Conditional random field based models [16, 31, 12, 8, 19] were used for long time. Various unary and pair-wise terms were discussed [18, 31]. High order terms [15, 29, 14, 20, 21] were also employed. Ladicky *et al.* [20] used a sliding window object detector to generate the score map for each pixel. This method needs much computation especially when the detector has to evaluate thousands of sliding windows. Also the large number of object candidates may bring false alarms.

In recent years, with the immense development of deep convolutional networks [17, 32], semantic segmentation has achieved great success. Early methods [6, 27, 25] resorted to superpixels. Farabet *et al.* [6] applied a deep convolutional network to learning multi-scale hierarchical representation for superpixels and used the feature and the classification score to parse a tree hierarchically. Pinheiro and Collobert [27] used a recurrent network to merge the superpixels represented by low level features. These methods are influenced by the quality of superpixels.

Starting from [7], which classifies segment proposals by objectiveness via state-of-the-art image classification models, Dai *et al.* [3] extended it using spatial pyramid pooling. Hariharan *et al.* [11] proposed simultaneous detection and segmentation. Albeit great improvement in performance, the computation is still heavy. The candidate segments are built from bottom-up. It may cause serious problems when initial segments are wrong.

Another line of research investigates the fully convolutional network [24]. The share of convolutional layers can reduce running time. Chen *et al.* [2] improved the performance by reducing the network stride with their "hole" method. It considers a fixed size receptive field and does not involve object level clues for faraway object parts.

To optimize the local boundary segmentation labels, conditional random fields were incorporated in a post process [2]. Methods in [33, 30] merged the CRF model into the network for joint training. These approaches handle local mistakes. They however could go wrong when a large part of a region is with incorrect labels.

With a large amount of unlabeled or weakly labeled data, semi-supervised setting was considered. In [26, 4], a semi-supervised training method was used. The model is trained using the VOC supervised [5] and COCO weakly supervised [23] data, and iteratively updates the mask or the label of the weakly supervised data for future training. This method is a promising direction for semantic segmentation, which avoids labor-intensive data annotation.

3. Object Regularized Semantic Segmentation

Given an image $\boldsymbol{I} \in \mathbb{R}^{m\times n}$, we use i to index pixels. x_i is the semantic label for pixel i. Our regularized semantic segmentation framework is formulated as

$$\begin{aligned} E(\boldsymbol{x},\boldsymbol{I}) = \sum_i \phi_1(x_i,\boldsymbol{I}) + \sum_{c\in\mathcal{C}}\sum_{i\in c}\phi_2(x_i,\boldsymbol{I}) \\ + \sum_{i,j\in\mathcal{N}}\phi_3(x_i,x_j,\boldsymbol{I}), \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the vector containing all labels in the image. For notation simplicity, we omit the dependency on $\boldsymbol{I}$, e.g. $E(\boldsymbol{x},\boldsymbol{I})$ is denoted as $E(\boldsymbol{x})$.

The first term $\phi_1(x_i)$ indicates the unary potential for pixel i, which will be introduced in Sec. 3.1. The second term $\phi_2(x_i)$ is our new object potential, the major contribution in our system. It will be elaborated on in Sec. 3.2. The third term $\phi_3(x_i,x_j)$ is a local appearance potential, with its definition in Sec. 3.3. These three terms work collaboratively for the final semantic segmentation prediction.

3.1. Unary Potential

Our unary term $\phi_1(x_i)$ aims to classify the region centered at pixel i. We resort to state-of-the-art image classification system [32, 24] and define this term on the output of a fully convolutional network as

$$\sum_i \phi_1(x_i) = -\sum_i \log(P(x_i|\boldsymbol{I})), \tag{2}$$

where $P(x_i|\boldsymbol{I})$ is the network softmax prediction for pixel i with label x_i. The $-\log(\cdot)$ operator maps maximization of a probability to minimization of the energy term.

The fully convolutional network we adopt is similar to the one presented in [32]. It keeps the image classification record for next-stage prediction in pixel labeling. The fully convolutional network [24] has the property that a fully connected layer equals to a 1×1 convolutional layer, and takes a large input to produce dense prediction.

To reduce the stride and produce denser prediction, we change the stride of the last two max pooling layers from 2 to 1, and use a modified `im2col` to produce input to next layer to stabilize the size of the receptive field. Specifically, we modify the `im2col` function to select each feature vector with a stride of 2. The final map uses 4×4 times channel number as the input channel number. The resulting receptive field of this network is with size 198×198, which is comparable to the original 224×224 receptive field. The stride size becomes 8.

This approach is similar to the hole method [2]. A large receptive field ensures class-specific classification decision. With our denser stride and comparable receptive field generation, unary prediction is improved. The final score map then goes into an upsampling layer [24] to reach the scale of input. We validate the steps in Section 5.

3.2. Object Potential

Our object potential works complementarily with the unary term to provide object-level information. It benefits the flexible receptive field formed on objects, which previously cannot be efficiently handled via the unary fully convolutional network. This potential makes traditionally confusing object parts, such as those of cat and dog, better handled under the global view on objects. Moreover, it avoids heavy computation [7] partly due to the small size of object proposals. Our object potential term is formulated as

$$\sum_{c\in\mathcal{C}}\sum_{i\in c}\phi_2(x_i) = \sum_{c\in\mathcal{C}}\sum_{i\in c}\Big[-\mathbb{I}_c(x_i,\tau)\log(s^c_{x_i})\Big], \tag{3}$$

where $c \in \mathcal{C}$ indexes an object clique. Each object clique contains a group of pixels with their semantic labels. Note that each pixel can belong to several cliques, since two adjacent object cliques can potentially form a new clique.

$\mathbb{I}_c(x_i,\tau)$ is an indicator function, which is non-zero when pixel i belongs to the c-th object clique under the proposal tree τ. We will give more details about τ later. $s^c_{x_i}$ is the x_i-th element in $\boldsymbol{s}^c$, where $\boldsymbol{s}^c$ is the probability vector for object proposal c predicted by an object detection system. $\boldsymbol{s}^c$ is thus a N-dimension vector for N-class semantic segmentation. Note that the group of pixels inside object clique $\boldsymbol{c}$ share the same object probability vector $\boldsymbol{s}^c$.

With the object potential defined in Eq. (3), a clique with high object confidence tends to predict its corresponding object label as the pixel label. In the following, we explain generation of object clique set $\mathcal{C}$, setting object probability vector $\boldsymbol{s}_c$, and clique indicator function $\mathbb{I}_c(\cdot)$ definition.

Object Clique Generation Object proposal generation is crucial for our system since it upper bounds the capability of our object potentials.

We start with the label map output from our fully convolutional network (FCN), which generates the class label for the minimum unary potential in Eq. (2), as shown in Fig. 2(b). The FCN label map gives us a rough indication of existence of the object, which serves as an initial seed for our proposal clique set $\mathcal{C}$. We build a proposal tree τ on top of it to identify the object cliques.

In particular, given an image $\boldsymbol{I}$ and the corresponding label map $\boldsymbol{L}$, we first include neighboring regions with the same label as cliques. Note that we remove the background label region, as well as the regions whose sizes are smaller than 625 pixels, since the scores for such regions are not reliable. These initial cliques are then hierarchically merged to generate new cliques based on the connectivity and similarity of regions until all connected regions are grouped.

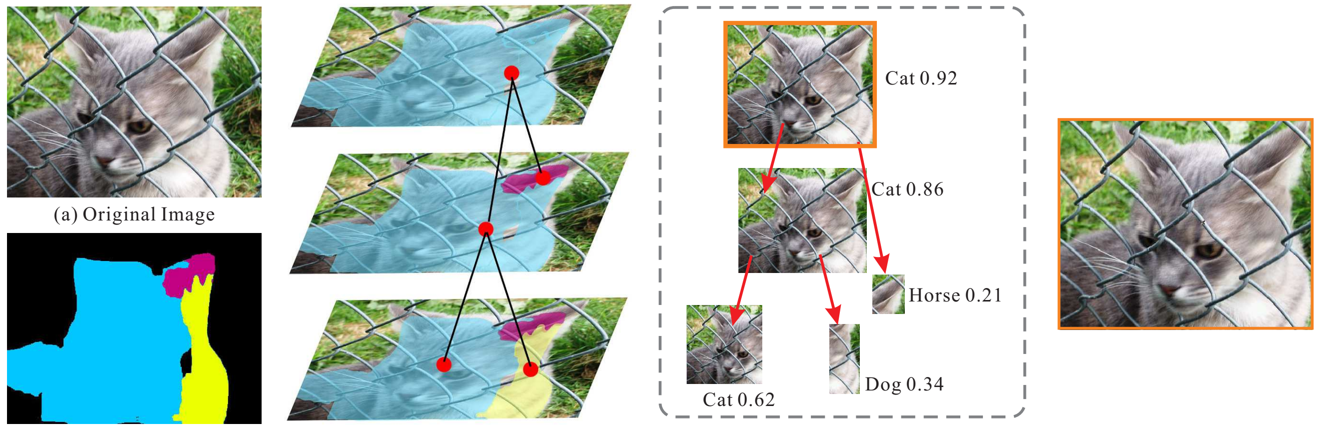
If two or more regions are adjacent to each other, we resort to the object probability vector $\boldsymbol{s}^c$ to group similar ones first. The definition of $\boldsymbol{s}^c$ will be introduced later. This hierarchical grouping strategy results in the proposal tree set denoted as τ, where each tree represents the merging process of object cliques, and each node is an object clique c. The grouping strategy is illustrated in Fig. 2(c) and sketched in Algorithm 1.

Our object clique generation strategy produces 4 proposals per image on average, which defines the number of objects for the detection system to get the object probability vector. Compared to previous state-of-the-art pipelines, which give about 2,000 proposals per image [7, 3], ours is significantly more efficient and contains less false proposals. We verified that our label map covers about 84.7% objects in the VOC 2012 dataset, which is a large and reasonable ratio compared to previous ones. Those missing object cliques include small, distant, and blurred objects, which are very difficult to be recognized in the region level.

Object Probability Construction Equipped with the object cliques $c \in \mathcal{C}$, our framework constructs an object probability vector $\boldsymbol{s}^c$, which determines the object-level guidance in the overall process. Our object probability vector is defined as

$$\boldsymbol{s}^c = (\mathcal{W}\boldsymbol{l}^c)\cdot\boldsymbol{d}^c, \tag{4}$$

where $\mathcal{W}\boldsymbol{l}^c$ is a co-occurrence prior; $\boldsymbol{d}^c$ is the object confidence score; and $\cdot$ denotes element-wise multiplication.



(a) Original Image

(b) Label Map

(c) Hierarchical Tree

(d) Object Cliques and Max Object Scores

(e) Selected Object Clique

Figure 2. Illustration of the object potential. (a) is the input image. (b) is the label map where each color represents a class. (c) shows the hierarchical tree τ built on the label map. (d) shows a few object clique bounding boxes in the set $\mathcal{C}$. The maximum object score $s^c_{x_i}$ for each clique is listed on the right for clique identification. After parsing the clique tree with the object probability vector $\boldsymbol{s}^c$, only the root clique is enabled as the final object potential.

The co-occurrence prior $\mathcal{W}\boldsymbol{l}^c$ provides context information based on training data statistics. For example, a horse can be beside a person. But a horse has a very low probability to be co-occurrent with a dinning-table. Therefore, if the FCN label map predicts a region as a horse, the co-occurrence prior imposes the high probability to both the horse and person, but not the dinning-table.

Accordingly, $\mathcal{W}$ is the co-occurrence matrix obtained by counting the co-occurrent object information. $\boldsymbol{l}^c$ is a binary vector indicating the existence of a class in object clique c on the label map. The multiplication of $\mathcal{W}$ and $\boldsymbol{l}^c$ gives an N-dimension vector for the N-class labeling problem. Its i-th element defines the probability that FCN label map reports the i-th label based on the co-occurrent information and its current label configuration $\boldsymbol{l}^c$.

The object confidence score $\boldsymbol{d}^c$ can be understood as providing the global view of objects, which avoids inaccuracy caused by the fixed receptive field in the unary term via FCN. It is defined via an object detection system [7], which extracts feature by the fine-tuned VGG model [32], and feeds into an SVM model for prediction. To calibrate the SVM score into a probability distribution to fit our model in Eq. (3), we use the method of [28] to map it to a sigmoid distribution likelihood as

$$\boldsymbol{d}^c = 1 + \exp(-(\boldsymbol{\alpha} \cdot \boldsymbol{f}^c + \boldsymbol{\beta})), \tag{5}$$

where $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}^{N \times 1}$ are the calibration parameters. They are learned on training data with the ground-truth label vector $\boldsymbol{d}^c$.

Since in the object potential part, we only focus on foreground objects, both $\mathcal{W}\boldsymbol{l}^c$ and $\boldsymbol{d}^c$ are with $N-1$ dimensions if the problem contains N classes including background. We append an additional 1 to the probability vector $\boldsymbol{s}^c$ to ease optimization without introducing any overhead.

Algorithm 1 Object Clique Generation and Probability Construction

```
Input: FCN heat map $\mathcal{H}$;
Procedure:
Initialize $\mathcal{C}$ with all foreground object regions in $\mathcal{H}$;
Initialize $\boldsymbol{s}^c$ via Eq. (4) $\forall c \in \mathcal{C}$;
while adjacent foreground regions exist do
    Select 2 adjacent objects in $\mathcal{C}$ with most similiar $\boldsymbol{s}^c$;
    Merge the two regions as a new object in $\mathcal{C}$;
    Remove previous two regions from $\mathcal{C}$;
    Calculate $\boldsymbol{s}^c$ via Eq. (4);
end while
Output: All object cliques $c$ in $\mathcal{C}$ and its probability vector
    $\boldsymbol{s}^c$; the clique merging tree $\tau$.
```

We summarize the object clique generation and the probability vector construction process in Alg. 1. Our object probability vector has the following advantages. First, the context information is incorporated via the co-occurrence prior to utilize big data. Second, the detection-based confidence vector gives us a different view of objects from top down, which serves as a vital complement of the unary term via FCN. Moreover, with the elegant scale of object clique set and its high object recall, we only need to test a few candidates, which is much more efficient than previous segment-proposal-based methods [3, 11] without sacrificing accuracy.

Object Clique Selection We note not all cliques represent reasonable objects. To alleviate the adverse effect caused by false alarms, we define our object clique indicator function

Algorithm 2 Object Clique Selection via Message Passing

```
Input:
    Clique merging tree $\tau$ defined in Alg. 1;
    object probability vector $\boldsymbol{s}^c$ for all nodes $c$ in $\tau$;
    parent-child threshold $\gamma$.
Procedure:
    Initialize $m = \arg\max_j s_j^c,\ \ \forall\, c \in \tau$;
    Initialize $V$ as node list in $\tau$ in a breath-first search;
    Initialize $V_t = V$;
    Initialize $\mathbb{I}_c(j, \tau) = 0,\ \forall j, c$;
    for $p \in V_t$ do
        Construct $V_s$ as the child node of $p$;
        for $s \in V_s$ do
            if $s_m^p > s_m^s - \gamma$ then
                $\boldsymbol{s}^s = \boldsymbol{s}^p$;
                remove $s$ and all its children from $V_t$;
            end if
        end for
        Remove $p$ if all its children $s$ are left in $V_t$;
    end for
    Assign the leaf node in $V$ with the probability vector of
        its lowest ancestor in $V_t$.
Output: $\mathbb{I}_c(m, \tau) = 1$ for $c \in V_t$;
        Probability vector of each leaf node.
```

as

$$\mathbb{I}_c(x_i, \tau) = \begin{cases} 1 & \text{if } s_{x_i}^c > s_{x_i}^p + \gamma \cap s_{x_i}^c > s_{x_i}^s - \gamma,\ \ \forall p, s, \\ & \quad \cap\, s_{x_i}^c > s_j^c,\ \ \forall j \neq x_i; \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

where s_j^c is the j-th element in object probability vector $\boldsymbol{s}^c$. Thus only the predicted object label by the detection system has the object potential. $p, s \in \mathcal{C}$ are the object clique indices in the clique proposal tree τ. p denotes all possible ancestor nodes of c in the object clique tree τ. s can be all offspring nodes of c inside the clique tree τ. γ is a threshold parameter. It favors large cliques, since the top node is chosen unless the probability of its child node is larger and the gap is over a threshold γ.

Such indicator function can be calculated by traversing nodes and passing messages in the tree from top down. We use Alg. 2 to summarize the process. Our object potential gives additional confidence for object regions in its respective object category. Thus, if a misclassified region by the unary term predicts 'dog' while our object potential gives a high score as 'cat' by detection, the system can finally select the 'cat' label by merging the scores from unary and object potentials.

3.3. Local Appearance Potential

The fully convolutional network (FCN) consists of several spatial-invariant operations for object-level representation, such as max pooling and convolution layers, which make the score map generated by the FCN system difficult to preserve the accurate object contour. The object potential is based on connecting regions provided by FCN, where the contour is still not sufficiently accurate. Our pairwise potential is similar to the CRF model proposed in [2], expressed as

$$\begin{aligned} \phi_3(x_i, x_j) = \mathbb{I}(x_i, x_j)\Big(&\lambda_1 \exp(-\frac{\|p_i - p_j\|^2}{2\theta_\alpha^2}) \\ &+ \lambda_2 \exp(-\frac{\|p_i - p_j\|^2}{2\theta_\beta^2} - \frac{\|I_i - I_j\|^2}{2\theta_\gamma^2})\Big), \end{aligned} \tag{7}$$

where $\mathbb{I}(x_i, x_j)$ is an indicator function. It equals to 1 if $x_i \neq x_j$ and 0 otherwise. p_i is the position in x- and y-directions of pixel i. I_i denotes the RGB value vector of pixel i. θ_α, θ_β, and θ_γ are the parameters controlling the weight of each term. The first term favors close pixels while the second term gives large confidence to close pixels with similar local appearance. λ_1 and λ_2 balances the two parts.

With the local appearance potential, close pixels with similar local appearance tend to have the same label, thus further correcting local-boundary inaccuracy. Note that our new object potential presented in Sec. 3.2 facilitated by top-down object-level information can contrarily correct labels in larger areas. So the two conditions, i.e., ϕ_2 and ϕ_3, work in synergy to update and improve both larger-area and local boundary labeling mistakes.

4. Optimization

Our final object regularized semantic segmentation framework predicts a pixel-wise label map $\boldsymbol{x}$ defined as $\arg\min_{\boldsymbol{x}} E(\boldsymbol{x})$. where $E(\boldsymbol{x})$ is given in Eq. (1). The problem is NP-hard. Following [31, 22], we adopt a stepwise optimization approach to approximate it.

The fully convolutional network (FCN) is pre-trained using off-the-shelf SGD algorithm following [24]. Thus the unary term by the FCN network is computed once during optimization. To optimize the object potential and local appearance potential, we resort to the following two steps.

Solve for Object Potential Given the initial label configuration, we construct the object clique set and its corresponding probability following Alg. 1. Then we parse the object clique generation tree to get the clique identification function in Eq. (6) via a message parsing algorithm in Alg. 2. In this step, each pixel gets a new score by combing the score map of the unary potential and the object potential.

Algorithm 3 Object Regularized Semantic Segmentation

```
Input:
    FCN label map for unary term;
    Maximum iteration number $T$;
Procedure:
    Initialize $\phi_2(x_i) = 0$ for all $i$;
    for $i \in [1, T]$ do
        Solve Eq. (8) via [16];
        Use Alg. 1 to build object clique set $\mathcal{C}$ and its probability vector $\boldsymbol{s}^c$;
        Use Alg. 2 to construct clique indicator in Eq. (6);
        Update $\Phi(x_i)$ for each $i$ via Eq. (9).
    end for
Output: Label map $\boldsymbol{x}$.
```

Solve for Local Appearance Potential In this round, with the updated pixel confidence combined by unary potential and object potential, the objective function is formulated as

$$\min_{\boldsymbol{x}} \sum_i \Phi(x_i) + \sum_{i,j} \phi_3(x_i, x_j). \tag{8}$$

$\Phi(x_i)$ is the combined confidence from Eqs. (2) and (3) for pixel i, expressed as

$$\Phi(x_i) = \phi_1(x_i) + \sum_{c \in \mathcal{C}} \mathbb{I}(i \in c)\phi_2(x_i) \tag{9}$$

where $\mathbb{I}(i \in c)$ is an indicator function which equals to 1 when pixel $i \in c$. Thus, Eq. (8) becomes a traditional CRF problem and can be solved following [16], which uses mean-field approximation and takes 0.5 second to process an image. The overall optimization procedure is summarized in Alg 3.

5. Experiments

Dataset We evaluate our method on PASCAL VOC 2012 segmentation benchmark [5]. The number of training, validation and testing images are 1,464, 1,449 and 1,456 respectively. Following the scheme of [2], we also merge the additional 9,118 annotated images from Hariharan *et al.* [9] into the training set. This dataset is a standard dataset for semantic segmentation. It contains 20 object classes and 1 background class.

Unary Potential with Fully Convolutional Network Our unary potential in Eq. (2) is obtained from the prediction of a fully convolutional network. We have fine-tuned this network on the 10,582 training data initialized by the VGG-16 model [32]. The overall network is implemented based on Caffe [13].

When fine-tuning the model, the training images are resized, so that the short side of the image is with 256 pixels. We adopt the conventional data augmentation strategy with random cropping and mirroring. Dropout is used the same as the original VGG model. The stride for max pooling is reduced to 1, with a modified `im2col` to preserve a stable receptive field as detailed in Sec. 3.1. The initial learning rate is set to 0.001. It decays with a factor of 0.1 after every 4000 iterations until $1e^{-7}$.

Methods	mean IoU%
Baseline FCN	62.48
FCN+Local Appearance	64.45
FCN+Object Potential	64.07
FCN+Object Potential (w/o co-occurence prior)	63.16

Table 1. Comparison of our baseline FCN model, FCN with local appearance potential as in [2], and FCN with object regularization respectively. We implement the local appearance potential [2].

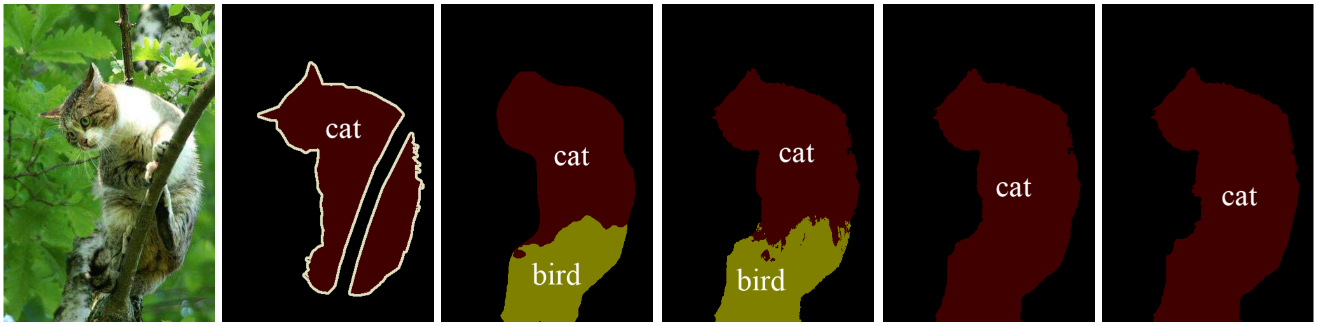
We evaluate the performance of our network on the VOC 2012 validation set in terms of average per class predicting intersection-over-union (IoU) across the 21 classes. Our baseline model with stride reduction and upsampling layer [24] achieves 62.48% mean IoU. The stride reduction strategy improves the network performance by 4.0%. The upsampling layer only improves it by 0.2%.

Object Regularized Semantic Segmentation All the parameters of our approach, including α and β in Eq. (5), γ in Eq. (6), and θ_α, θ_β, θ_γ, and λ_1, λ_2 in Eq. (7) are obtained from cross-validation on the validation set following the strategy of [2, 4]. The input to the object detection system is the object proposal bounding box with 16-pixel padding following that of [7].

The performance of our baseline FCN model is listed in Table 1, which is only the unary term in Eq. (1). We also include the model with unary potential and local appearance potential, and the model with unary potential and object potential, respectively. The results are tabulated in Table 1. Note that our implemented local appearance potential yields performance improvement 2.21% compared to the 3.94% reported in [2]. It is possibly because we cannot find the best parameter values. Our new object regularization yields additional improvement in our system nevertheless.

Our final model is optimized in an iterative manner as shown in Alg. 3. We fix the output of the FCN system and iteratively optimize the object and local appearance potentials. We evaluate the performance on the validation set. After 3 iterations, the system ceases the update, as shown in Table 2. In following experiments, we use two iterations for the sake of computation efficiency.

We show iterative update in Fig. 3. The local appearance term refines the object boundary. But it does not correct large errors caused by FCN. Our object regularization contrarily updates labels for these large regions.



(a) Original Image (b) Groundtruth (c) FCN (d) with local appearance (e) with object potential (f) Our final result

Figure 3. Visual illustration of results produced in iterations. Our object potential can correct a few large errors resulted from FCN.

Iterations	mean IoU%
Baseline FCN	62.48
Iteration 1 with Local Appearance	64.45
Iteration 1 with Object Potential	66.67
Iteration 2 with Local Appearance	66.95
Iteration 2 with Object Potential	66.98
Iteration 3 with Local Appearance	66.96
Iteration 3 with Object Potential	66.99

Table 2. Performance of our object regularized semantic segmentation in iterations.

Methods	mean IoU%
Original $\times 0.8$	62.60
Original $\times 1$	62.48
Original $\times 1.2$	61.04
Average of three scales	62.57
Original with Object Potential	64.07

Table 3. Comparison with the multi-scale strategy. We test three scales in our experiments, i.e., $\{0.8, 1, 1.2\}$ times the original scale. Our object potential strategy is still better than average of the three scales.

Comparison with Multiscale Strategy Multiscale training and testing can also correct errors caused by the fixed receptive in the FCN-system. By scaling the image, the receptive field of the network can be updated with respect to the original image resolution. We compare our method with this strategy.

We use three scales as suggested in other multiscale methods [6]. The images are scaled with ratios $\{0.8, 1, 1.2\}$ w.r.t. their original resolutions. Then we average the three scale outputs and get the final results from the network. The performance of the three scales is shown in Table 3.

As shown above, our object regularization improves semantic segmentation performance from 62.48% to 64.07%. In comparison, combining the three different scales of FCN yields the change from 62.48% to 62.57%. This shows that our regularization is more effective to handle the flexible receptive field.

Method	Proposal	CNN	CRF
SDS [11]	34.3s [1]	17.9s	-
CFM [3]	34.3s [1]	2.10s	-
FCN-8s [24]	-	0.21s	-
DeepLab [2]	-	0.13s	0.50s
Ours-crop	-	1.77s	0.52s

Table 4. Running-time comparison of different methods on one image with the original image resolution.

Efficiency Comparison In the following, we compare the running time. The result is shown in Table 4. Our method is evaluated on a NVIDIA Tesla K40 GPU. Running time of other methods are quoted from respective papers.

Compared to SDS [11], the CFM [3] system reduces the running time using the spatial pyramid pooling, which computes features for the whole image only once. FCN-8s [24], DeepLab [2] and our method save time using fully convolution networks that reuse the convolution features. Object proposal methods take time to generate object proposals, where MCG can be used to gain reasonable performance.

Performance Comparison We compare our method with others, including Hypercolumn [10], CFM [3], FCN [24], TTI-zoomout [25], and DeepLab[2], on PASCAL VOC 2012 test set. The best results in Table 5 are mostly those of [2] with the large receptive field model[1].

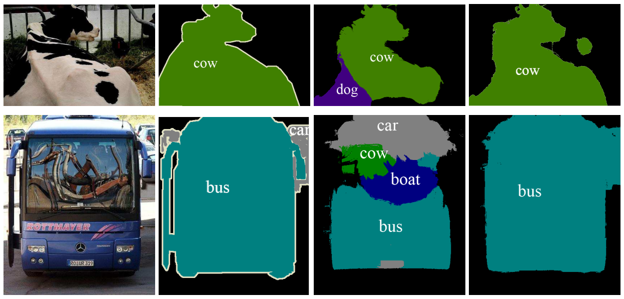
Our method is effective in discriminating among objects that could easily confuse previous systems. A visual comparison in Fig. 4 shows that our object regularized semantic segmentation successfully separates the cow and bus cases, which are difficult for that of [2]. The CRF method works better on objects with very complex contours such as bird, chair and plant. Our implemented CRF does not reach the performance reported in [2], which has been explained above.

Further, we compare our method with bottom-up segmentation [3]. Results are shown in Fig. 5. The errors in

[1] https://bitbucket.org/deeplab/deeplab-public/

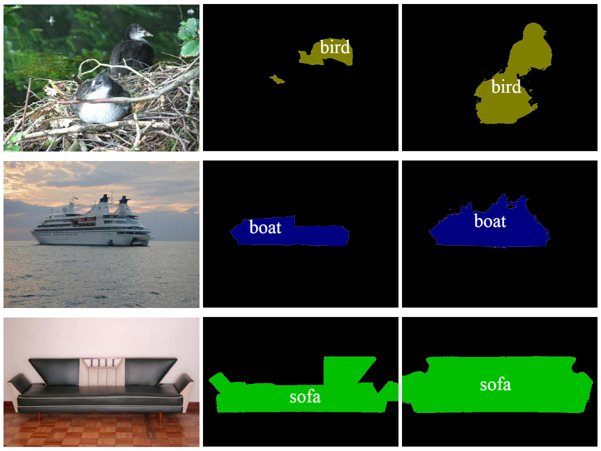
Methods	IoU	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
CFM [3]	61.8	75.7	26.7	69.5	48.8	65.6	81.0	69.2	73.3	30.0	68.7	51.5	69.1	68.1	71.7	67.5	50.4	66.5	44.4	58.9	53.5
FCN-8s [24]	62.2	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1
Hyper [10]	62.6	68.7	33.5	69.8	51.3	70.2	81.1	71.9	74.9	23.9	60.6	46.9	72.1	68.3	74.5	72.9	52.6	64.4	45.4	64.9	57.4
TTI [25]	69.6	85.6	37.3	83.2	62.5	66	85.1	80.7	84.9	27.2	73.3	57.5	78.1	79.2	81.1	77.1	53.6	74	49.2	71.7	63.3
DeepLab [2]	66.4	78.2	51.3	74.2	59.2	60.2	82.3	75.7	78.3	26.7	66.6	54.5	73.9	68.4	78.8	76.8	52.4	74.8	46.4	66.0	55.4
DeepLab+CRF [2]	70.9	84.0	53.7	80.4	63.1	64.6	85.2	78.4	82.5	29.1	73.8	60.0	79.1	75.0	82.4	80.4	58.1	80.1	50.7	71.9	63.4
Ours	66.5	79.2	33.8	75.4	48.5	64.8	85.1	79.3	81.5	24.8	77.7	51.5	78.1	76.3	77.8	77.2	46.0	78.2	42.5	71.1	55.4
DeepLab+Our CRF	71.2	80.0	53.8	80.8	62.5	64.7	87.0	78.5	83.0	29.0	82.0	60.3	76.3	78.4	83.0	79.8	57.0	80.0	53.1	70.1	63.1

Table 5. Performance Comparison on Pascal VOC 2012 test set.



(a) Input (b) GT (c) Results [2] (d) Our results

Figure 4. Result comparison.

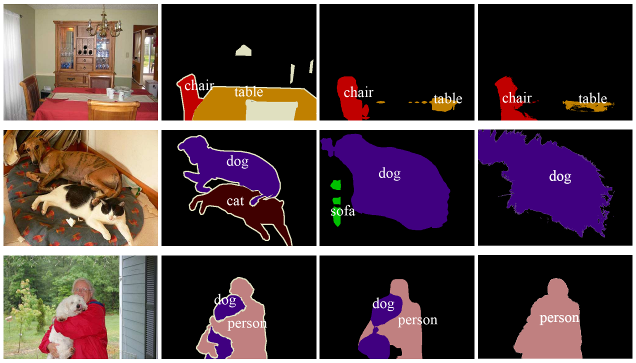


(a) Input (b) Results [3] (d) Our results

Figure 5. Result comparison.

the results of [3] are caused by erroneous initial object segmentation. Our method alleviates this problem by resorting to the flexible FCN system with the object and local appearance potentials, which does not need to perform explicit segmentation.

Failure Case Analysis Our failure cases can be categorized into three types, i.e., missing objects in the FCN map, merged objects, and occluded objects. The incomplete table in the first example of Fig. 6 is due to the foreground object misclassified as background in FCN. The second example contains two objects merged together by the FCN system.



(a) Input (b) Ground truth (c) FCN map (d) Final result

Figure 6. Three failure examples.

These errors cannot be well handled by our current framework. Occlusion failure arises in the third example. Our object scoring strategy favors large objects to help general image segmentation. In case of serious occlusion, the bottom prediction of small objects is not confident. This problem might be alleviated by back-propagating the error to the FCN system, which will be part of our future work.

6. Conclusion

We have presented a novel object potential for semantic segmentation. It solves the problem that the original fully convolutional network lacks top-down object-level information. Our system enjoys the efficiency benefit yielded from solving fully convolutional networks. Our proposed message passing algorithm can efficiently identify object scores. The object potential is also functionally complementary to current local appearance methods, as demonstrated in our experiments. Our future work lies in extending our method to semi-supervised or unsupervised configuration.

Acknowledgements

This research is supported by the Research Grant Council of the Hong Kong Special Administrative Region under grant number 413113.

References

[1] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, pages 328–335, 2014.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[3] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. *arXiv preprint arXiv:1412.1283*, 2014.

[4] J. Dai, K. He, and J. Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. *arXiv preprint arXiv:1503.01640*, 2015.

[5] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE TPAMI*, 35(8):1915–1929, 2013.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[8] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, pages 1–8, 2009.

[9] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, pages 991–998, 2011.

[10] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. *arXiv preprint arXiv:1411.5752*, 2014.

[11] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, pages 297–312. 2014.

[12] X. He and S. Gould. An exemplar-based crf for multi-instance object segmentation. In *CVPR*, pages 296–303, 2014.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, pages 675–678, 2014.

[14] P. Kohli, M. P. Kumar, and P. H. Torr. P3 & beyond: Solving energies with higher order cliques. In *CVPR*, pages 1–8, 2007.

[15] P. Kohli, P. H. Torr, et al. Robust higher order potentials for enforcing label consistency. *IJCV*, 82(3):302–324, 2009.

[16] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *NIPS*, pages 109–117, 2011.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[18] L. Ladicky, C. Russell, P. Kohli, and P. H. Torr. Associative hierarchical crfs for object class image segmentation. In *ICCV*, pages 739–746, 2009.

[19] L. Ladickỳ, C. Russell, P. Kohli, and P. H. Torr. Inference methods for crfs with co-occurrence statistics. *IJCV*, 103(2):213–225, 2013.

[20] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and crfs. In *ECCV*, pages 424–437. 2010.

[21] Y. Li, D. Tarlow, and R. Zemel. Exploring compositional high order pattern potentials for structured output learning. In *CVPR*, pages 49–56, 2013.

[22] G. Lin, C. Shen, I. Reid, et al. Efficient piecewise training of deep structured models for semantic segmentation. *arXiv preprint arXiv:1504.01013*, 2015.

[23] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. 2014.

[24] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv preprint arXiv:1411.4038*, 2014.

[25] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. *arXiv preprint arXiv:1412.0774*, 2014.

[26] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly-and semi-supervised learning of a dcnn for semantic image segmentation. *arXiv preprint arXiv:1502.02734*, 2015.

[27] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene parsing. *arXiv preprint arXiv:1306.2795*, 2013.

[28] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[29] S. Ramalingam, P. Kohli, K. Alahari, and P. H. Torr. Exact inference in multi-label crfs with higher order cliques. In *CVPR*, pages 1–8, 2008.

[30] A. G. Schwing and R. Urtasun. Fully connected deep structured networks. *arXiv preprint arXiv:1503.02351*, 2015.

[31] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 81(1):2–23, 2009.

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[33] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. *arXiv preprint arXiv:1502.03240*, 2015.