Proposal-Free Network for Instance-Level Object Segmentation

Xiaodan Liang, Liang Lin, Yunchao Wei, Xiaohui Shen, Jianchao Yang, and Shuicheng Yan

Abstract—Instance-level object segmentation is an important yet under-explored task. Most of state-of-the-art methods rely on region proposal methods to extract candidate segments and then utilize object classification to produce final results. Nonetheless, generating reliable region proposals itself is a quite challenging and unsolved task. In this work, we propose a Proposal-Free Network (PFN) to address the instance-level object segmentation problem, which outputs the numbers of instances of different categories and the pixel-level information on i) the coordinates of the instance bounding box each pixel belongs to, and ii) the confidences of different categories for each pixel, based on pixel-to-pixel deep convolutional neural network. All the outputs together, by using any off-the-shelf clustering method for simple post-processing, can naturally generate the ultimate instance-level object segmentation results. The whole PFN can be easily trained without the requirement of a proposal generation stage. Extensive evaluations on the challenging PASCAL VOC 2012 semantic segmentation benchmark demonstrate the effectiveness of the proposed PFN solution without relying on any proposal generation methods.

Index Terms—Instance-level object segmentation, semantic segmentation, region-proposal-free, convolutional neural network

1 Introduction

Over the past few decades, two of the most popular object recognition tasks, object detection and semantic segmentation, have received a lot of attention. The goal of object detection is to accurately predict the semantic category and the bounding box location for each object instance, which is a quite coarse localization. Different from object detection, the semantic segmentation task aims to assign the pixel-wise labels for each image but provides no indication of the object instances, such as the number of object instances and precise semantic region for any particular instance. In this work, we follow some of the recent works [1], [2], [3] and attempt to solve a more challenging task, instance-level object segmentation, which predicts the segmentation mask for each instance of each category. We suggest that the next generation of object recognition should provide a richer and more detailed parsing for each image by labeling each object instance with an accurate pixel-wise segmentation mask. This is particularly important for real-world applications such as image captioning, image retrieval, 3-D navigation and driver assistance, where describing a scene with detailed individual instance regions is potentially more informative than describing roughly with located object detections. However, instance-level object segmentation is very challenging due to high occlusion, diverse shape deformation and appearance patterns, obscured boundaries with respect to other instances and background clutters in real-world scenes. In addition, the exact number of instances of each category within an image is dramatically different.

Recently, tremendous advances in semantic segmentation [4], [5] and object detection [6], [7], [8] have been made relying on deep convolutional neural networks (DCNN) [9], [10]. Some previous works have been proposed to address instance-level object segmentation. In general, these previous methods take complicated pre-processing such as bottom-up region proposal generation [11], [12], [13], [14] or post-processing such as graphical inference as the requisite. Specifically, the recent two approaches, SDS [1] and the one proposed by Chen et al. [2], use the region proposal methods to first generate potential region proposals and then classify on these regions. After classification, post-processing such as non-maximum suppression (NMS) or Graph-cut inference, is used to refine the regions, eliminate duplicates and rescore these regions. Note that most region proposal techniques [11], [12], [14] typically generate thousands of potential regions, and take more than one second per image. Depending on the region proposal techniques, the common pipelines are often trained using several independent stages. These separate pipelines rely on independent techniques at each stage and the targets of the stages are significantly different.

- *X. Liang and L. Lin are with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, Guangdong Province 510275, China and also with Engineering Research Center for Advanced Computing Engineering Software of Ministry of Education, China. E-mail: xdliang328@gmail.com, linliang@ieee.org.*
- *Y. Wei is with the Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China. E-mail: wychao1987@gmail.com.*
- *X. Shen is with Adobe Research, San Jose, CA 95110-2704. E-mail: xshen@adobe.com.*
- *J. Yang is with Snapchat Research, Venice, CA 90291. E-mail: jiayang@adobe.com.*
- *S. Yan is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077. E-mail: eleyans@nus.edu.sg.*

Manuscript received 4 Sept. 2015; revised 17 Oct. 2017; accepted 14 Nov. 2017. Date of publication 22 Nov. 2017; date of current version 1 Nov. 2018.
(Corresponding author: Liang Lin.)
Recommended for acceptance by D. Ramanan.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2017.2775623

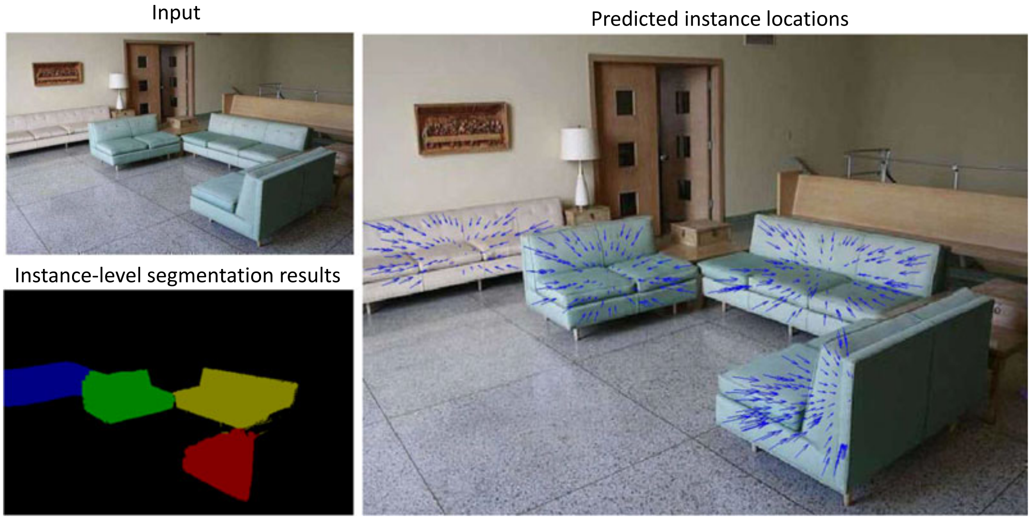

Input
Predicted instance locations
Instance-level segmentation results

Fig. 1. Exemplar instance-level object segmentation results. Different colors indicate the different object instances for each category. To better show the predicted locations of each instance, we plot velocity vectors starting from each pixel to its corresponding predicted instance center as shown by the arrow. Note that the pixels predicting similar object centers can be directly collected as one instance region. Best viewed in color.

For example, the region proposal methods try to maximize region recalls while the classification optimizes for single class accuracy.

In this paper, we propose a simple yet effective Proposal-Free Network (PFN) for solving the instance-level segmentation task. The motivation of the proposed network is illustrated in Fig. 1. The pixels belonging to the same instance can be naturally clustered as an instance. For simplicity, we use the term *instance locations* to denote the coordinates of the bounding box of the instance each pixel belongs to. We reformulate the instance-level segmentation task in the proposed network by directly inferring the regions of object instances from the global image context. The proposed PFN framework is shown in Fig. 2. To solve the semantic instance-level object segmentation task, three sub-tasks are addressed: category-level segmentation, instance location prediction for each pixel, and number prediction of instances for each category in the entire image.

First, the convolutional network is fine tuned based on the pre-trained VGG classification net [10] to predict the category-level segmentation. In this way, the domain-specific feature representation on semantic segmentation for each pixel can be learned.

Second, by fine-tuning on the category-level segmentation network, the instance locations for each pixel as well as the number of instances of each category are simultaneously predicted by the further updated instance-level segmentation network. The prediction target for each pixel is called as instance locations, which consists of four coordinates offsets of each pixel with respect to that of the top left corner and the bottom right corner of the bounding box of each instance. The precise prediction of instance locations is very crucial for segmenting the heavily occluded instances. To obtain more precise instance location prediction for each pixel, multi-scale prediction streams with individual supervision (i.e., multi-loss) are appended to jointly encode local details from the early, fine layers and the global semantic information from the deep, coarse layers. The feature maps from deep layers often focus on the global structure, but are insensitive to local boundaries and spatial displacement. In contrast, the feature maps from early layers can sense better the local detailed boundaries. The fusion layer combining multi-scale predictions is utilized before the final prediction layer.

Third, the number of instances of all categories are described with a real number vector and also regressed with euclidean loss in the instance-level segmentation network.

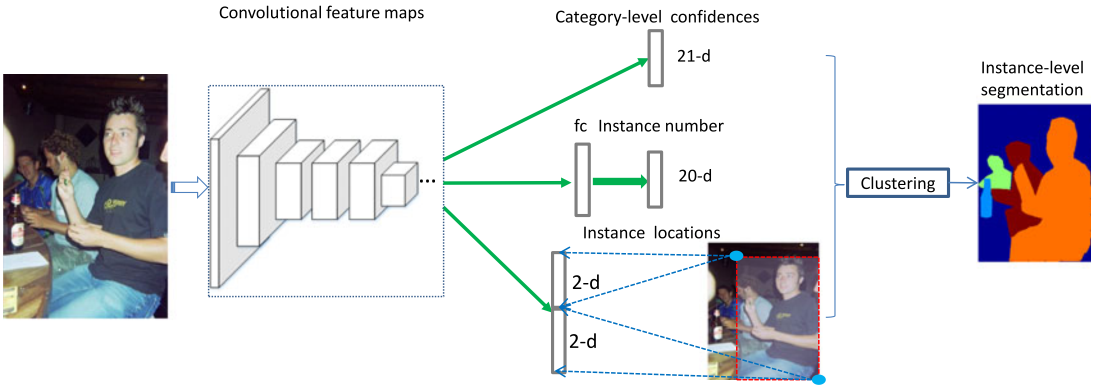

Convolutional feature maps
Category-level confidences
21-d
fc Instance number
20-d
Instance locations
2-d
2-d
Clustering
Instance-level segmentation

Fig. 2. An overview of our proposal-free network. The targets of our PFN include the number of instances of all categories, category-level confidences for each pixel and the coordinates of the instance bounding box that each pixel belongs to. The off-the-self clustering method can be utilized to generate final instance-level segmentation results.

Note that the number of instances embraces the category-level information (whether the instance of a specific category appears or not) and instance-level information (how many object instances appear for a specific category). Thus, the intermediate feature maps from the category-level segmentation network and the instance-level feature maps after the fusion layer from the instance-level segmentation network are concatenated together, which can be utilized to jointly predict the number of instances.

In the testing stage, the number of instances and pixel-level information including category-level confidences and coordinates of the instance bounding box each pixel belongs to, can together help generate the final instance-level segmentation results after clustering. Note that any off-the-self clustering method can be used for this simple post-processing, and the predicted number of instances specifies the exact number of clusters for the corresponding category.

Comprehensive evaluations and comparisons on the PASCAL VOC 2012 segmentation benchmark well demonstrate that the proposed proposal-free network yields results that significantly surpass all previous published methods. It should be noted that all previous works utilize the extra region proposal extraction algorithms to generate the region candidates and then feed these candidates into a classification network and complex post-processing steps. Instead, our PFN generates the instance-level segmentation results in a much simple and more straightforward way.

2 Related Work

Deep convolutional neural networks have achieved great success in object classification [9], [10], [15], [16], object detection [6], [7], [17] and object segmentation [1], [4], [18], [19]. In this section, we discuss the most relevant work on object detection, semantic segmentation and instance-level object segmentation.

Object Detection. Object detection aims to localize and recognize every object instance with a bounding box. The detection pipelines [6], [7], [20], [21] generally start from extracting a set of box proposals from input images and then identify the objects using classifiers or localizers. The box proposals are extracted either by the hand-crafted pipelines such as selective search [12], EdgeBox [13] or the designed convolutional neural network such as deep MultiBox [22] or region proposal network [6]. For instance, the region proposal network [6] simultaneously predicts object bounds and objectiveness scores to generate a batch of proposals and then uses the Fast R-CNN [23] for detection. Different from these prior work, Redmon et al. [7] first proposed a You Only Look Once (YOLO) pipeline that predicts bounding boxes and class probabilities directly from full images in one evaluation. Our work shares some similarities with YOLO, where the region proposal generation is discarded. However, our PFN is based on the intuition that the pixels inferring similar instance locations can be directly collected as a single instance region. The pixel-wise instance locations and the number of instances of each category are simultaneously optimized in one network. Finally, the fine-grained segmentation mask of each instance can be produced with our PFN instead of the coarse outputs depicted by the bounding boxes from YOLO.

Semantic Segmentation. The most recent progress in object segmentation [4], [18] was achieved by fine-tuning the pre-trained classification network with the ground-truth category-level masks. For instance, Dai et al. [4] estimated segmentation masks for training by extracting region proposals from the annotated boxes. Papandreou et al. [18] utilized the foreground/background segmentation methods to generate segmentation masks, and conditional random field inference is used to refine the segmentation results. Zheng et al. [5] formulated the conditional random fields as recurrent neural networks for dense semantic prediction. Different from the category-level prediction by these previous methods, our PFN targets at predicting the instance-level object segmentation that provides more powerful and informative predictions to enable the real-world vision applications. These previous pipelines using the pixel-wise cross-entropy loss for semantic segmentation cannot be directly utilized for instance-level segmentation because different instances are often distinguished by their spatial translations, and the output size of prediction maps cannot be constrained to a pre-determined number due to the uncertain maximal number of instances of all categories in all images.

Instance-Level Object Segmentation. Recently several approaches which tackle the instance-level object segmentation [1], [2], [3], [24], [25], [26], [27], [28], [29] have emerged. Most of the prior works utilize the region proposal methods as the requisite. For example, Hariharan et al. [1] classified region proposals using features extracted from both the bounding box and the region foreground with a jointly trained CNN. Similar to [1], Chen et al. [2] proposed to use the category-specific reasoning and shape prediction through exemplars to further refine the results after classifying the proposals. Dai et al. [28] trained to classify the region-level feature maps that are masked out based on region proposals. [24] designed a higher-order loss function to make an optimal cut in the hierarchical segmentation tree based on the region features. Other works have resorted to the object detection task to initialize the instance segmentation and the complex post-processing such as integer quadratic program [25] and probabilistic model [30] to further determine the instance segments. Hariharan et al. [26] defined the hypercolumn at a pixel as the vector of activations of all CNN units above that pixel to allow more precise localization. Dai et al. [27] proposed to use multi-task network cascades for differentiating instances, estimating masks, and categorizing objects. Ren et al. [29] introduced an integrated recurrent network with an attention mechanism, which sequentially employs a box proposal network, a segmentation network and a scoring network to obtain instance-level predictions.

These prior works based on region proposals are very complicated due to several pre-processing and post-processing steps. In addition, combining independent steps is not an optimal solution because the local and global context information cannot be incorporated together for inferring. In contrast, our PFN directly predicts pixel-wise instance location maps and uses a simple clustering technique to generate instance-level segmentation results. In particular, Zhang et al. [3] predicted depth-ordered instance labels of the image patch and then combined predictions into the final labeling via the Markov Random Field

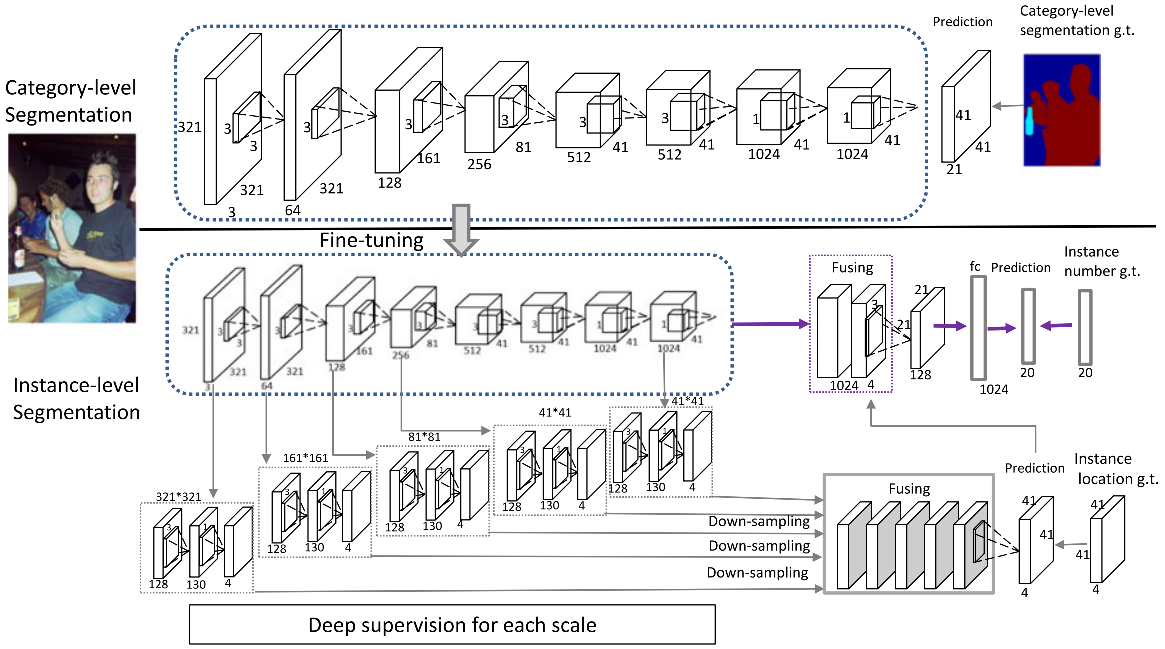


Fig. 3. The detailed network architecture and parameter setting of PFN. First, the category-level segmentation network is fine-tuned based on the pre-trained VGG-16 classification network. The cross-entropy loss is used for optimization. Second, the instance-level segmentation network that simultaneously predicts the number of instances of all categories and the instance location vector for each pixel is further fine-tuned. The multi-scale prediction streams (with different resolution and reception fields) are appended to the intermediate convolutional layers, and are then fused to generate ultimate instance location predictions. The regression loss is used during training. To predict the number of instances, the convolutional feature maps and the instance location maps are concatenated together for inference, and the euclidean loss is used. The two losses from two targets are jointly optimized for the whole network training.

inference. However, the number of instances to be present in each image patch is limited to be smaller than 6 (including background), which makes it not scalable for real-world images with an arbitrary number of possible object instances. Instead, our network predicts the number of instances in a totally data-driven way by the trained network, which can be naturally scalable and easily extended to other instance-level recognition tasks.

Recently, more research interests have been attracted to develop kinds of "proposal-free"-like pipelines [31], [32], [33], [34] that aim to generate instance-level object segmentation without relying on any proposal generation. Note that all these methods were proposed after the first submission of this paper. For example, Uhrig et al. [31] proposed to produce a semantic segmentation and instance-aware angle map that encoding the instance centroids. They also used ground-truth pixel-wise depth labeling for training. Bai et al. [32] learned the energy map of the watershed transform with the assumption that instances naturally correspond to basins in the energy map. [33] employed both semantic segmentation and object boundary prediction to separate instances. Liu et al. [34] trained sequential grouping networks to progressively group instance segments.

3 Proposal-Free Network

Fig. 3 shows the detailed network architecture of PFN. The category-level segmentation, instance locations for each pixel and the number of instances of all categories are taken as three targets during the PFN training.

3.1 Category-Level Segmentation Prediction

The proposed PFN is fine-tuned based on the publicly available pre-trained VGG 16-layer classification network [10] for the dense category-level segmentation task. We utilize the "DeepLab-CRF-LargeFOV" network structure as the basic presented in [35] due to its leading accuracy and competitive efficiency. The important convoluational filters are shown in the top row of Fig. 3, and other intermediate convolutional layers can be found in the published model file [35]. The reception field of the "DeepLab-CRF-LargeFOV" architecture is 224×224 with zero-padding, which enables effective prediction of the subsequent instance locations that requires the global image context for reasoning. For category-level segmentation, the 1000-way ImageNet classifier in the last layer of VGG-16 is replaced with $C+1$ confidence maps, where C is the number of categories. The loss function is the sum of pixel-wise cross-entropy in terms of the confidence maps (down-sampled by 8 compared to the original image). During testing, the fully-connected conditional random fields [35] are employed to generate more smooth and accurate segmentation maps.

This fine-tuned category-level network can generate semantic segmentation masks for the subsequent instance-level segmentation for each input image. Then the instance-level network is fine-tuned based on the category-level network, where the $C+1$ category-level predictions are eliminated. Note that we use two separate stages from optimizing category-level segmentation and instance-level segmentation. The intuition is that category-level segmentation prefers the prediction that is insensitive for different object instances of a specific category while instance-level segmentation aims to distinguish between individual instances. The motivations of two targets are significantly different. Therefore the convolutional feature maps, especially for the latter convolutional layers, cannot be shared. We verify the superiority of subsequently fine-tuning two separate networks for two tasks in the experiment. In addition, the performance on instance-level segmentation is much better when fine-tuning the instance-level network based on the category-level segmentation network compared to the original VGG-16. This may be because the category-level segmentation can provide a better start for parameter learning where the basic segmentation-aware convolutional filters have already been well learned.

3.2 Instance-Level Segmentation Prediction

The instance-level segmentation network takes an image with an arbitrary size as the input and outputs the corresponding instance locations for all the pixels and the number of instances belonging to each category.

Pixel-Wise Instance Location Prediction. For each image, the instance location vector of each pixel is defined as the bounding box information of its corresponding object instance that contains the pixel. The object instance s of a specific category can be identified by its top-left corner (l^x, l^y) and the bottom-right corner (r^x, r^y) of its surrounding bounding box, as illustrated in Fig. 2. For each pixel i with coordinates (p_i^x, p_i^y) belonging to the object instance s, we propose to predict the offsets between the coordinates of each pixel and the coordinates of its corresponding bounding box instead of directly predicting the coordinates of bounding boxes. The intuition is that due to various spatial displacements of objects in each image, the coordinates are more difficult to be predicted than the offsets between each pixel position and its corresponding object locations. The ground-truth instance location vector for each pixel is thus defined as the offsets $t_i = (\frac{l_s^x - p_i^x}{w_s}, \frac{l_s^y - p_i^y}{h_s}, \frac{r_s^x - p_i^x}{w_s}, \frac{r_s^y - p_i^y}{h_s})$, where w_s and h_s are the width and the height of the object instance s, respectively. With these definitions, we minimize an objective function to optimize the instance location which is inspired by the one used for Fast R-CNN [23]. Let t_i denote the predicted location vector and t_i^* the ground-truth location vector for each pixel i, respectively. The loss function ℓ^o can be defined as

$$\ell^o(t_i, t_i^*) = [k_i^* \geq 1] R(t_i - t_i^*), \tag{1}$$

where $k_i^* \in \{0, 1, 2, \ldots, C\}$ is the semantic label for the pixel i, and C is the total number of categories. R is the robust loss function (smooth-L_1) in [23]. The term $[k_i^* \geq 1] R(t_i - t_i^*)$ means the regression loss that is activated only for the foreground pixels and disabled for background pixels. The reason of using this filtered loss is that predicting the instance locations is only possible for foreground pixels which definitely belong to a specific instance.

Following the recent results of [35], we have also utilized the multi-scale prediction to increase the instance location prediction accuracy. As illustrated in Fig. 3, the five multi-scale prediction streams are attached to the input image, the output of each of the first three max pooling layers and the last convolutional layer (fc7) in the category-level segmentation network. For each stream, two layers (first layer: 128 convolutional filters, second layer: 130 convolutional filters) and deep supervision (i.e., individual loss for each stream) are utilized. The spatial padding for each convolutional layer is set so that the spatial resolution of feature maps is preserved. The multi-scale predictions from five streams are accordingly down-sampled and then concatenated to generate the fused feature maps (as the fusing layer in Fig. 3). Then the 1×1 convolutional filters are used to generate the final pixel-wise predictions. It should be noted that multi-scale predictions are with different spatial resolution and inferred under different reception fields. In this way, the fine local details (e.g., boundaries and local consistency) captured by early layers with higher resolution and the high-level semantic information from subsequent layers with lower resolution can jointly contribute to the final prediction.

Consider that the feature maps q^v of the vth convolutional layer are a three-dimensional array of size $h^v \times w^v \times d^v$, where h^v and w^v are spatial dimensions and d^v is the number of channels. The outputs q_{i_x, i_y}^{v+1} at the location (i_x, i_y) in the next layer can be computed by

$$x_{i_x, i_y}^{v+1} = f_b(\{q_{i_x + \delta_{i_x}, i_x + \delta_{i_y}}^v\}_{0 \leq \delta_{i_x}, \delta_{i_y} \leq b}), \tag{2}$$

where b is the kernel size and f_b is the convolutional filters. In PFN, q^{v+1} represents the final instance location prediction maps with four channels.

Suppose we have $M = 5$ multi-scale prediction streams, and each stream is associated with a regression loss $\ell_m^o(\cdot), m \in \{1, 2, \ldots, M\}$. For each image, the loss for the final prediction maps after fusing is denoted as $\ell_{\text{fuse}}^o(\cdot)$. The overall loss function for predicting pixel-wise instance locations then becomes

$$L^o(\mathbf{t}, \mathbf{t}^*) = \sum_{m=1}^{M} \sum_i \ell_m^o(t_i, t_i^*) + \sum_i \ell_{\text{fuse}}^o(t_i, t_i^*), \tag{3}$$

where $\mathbf{t} = \{t_i\}$ and $\mathbf{t}^* = \{t_i^*\}$ represent the predicted instance locations and ground-truth instance locations of all pixels, respectively.

Number Prediction of Instances. Another sub-task of PFN is to predict the numbers of instances of all categories. The number of instances of the input image that account for the object instances of each category naturally contains the category-level information and instance-level information. As shown in Fig. 3, the feature maps of the last convolutional layer from the previously trained category-level segmentation network and the instance location predictions are combined together to form the fused feature maps with $1024 + 4$ channels. These fused feature maps are then convolved with 3×3 convolutional filters and down-sampled with stride 2 to obtain the 128 feature maps. Then the fully-connected layer with 1,024 outputs is performed to generate the final C-dimensional number prediction maps of instances.

Given an input image I, we denote the number vector of instances of all C categories as $\mathbf{g} = [g_1, g_2, \ldots, g_C]$, where $g_c, c \in \{1, 2, \ldots, C\}$ represents the number of object instances of each category appearing in the image. Let $\mathbf{g}$ denote the predicted number vector of instances and $\mathbf{g}^*$ represent the ground-truth number vector of instances for each image, respectively. The loss function L^n is defined as

$$L^n(\mathbf{g}, \mathbf{g}^*) = \frac{1}{C} \sum_{c=1}^{C} ||g_c - g_c^*||^2. \tag{4}$$

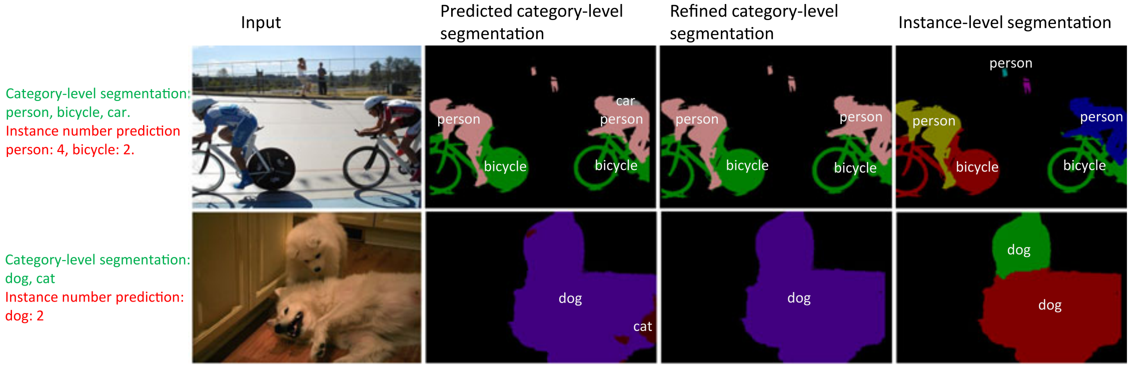

Input
Predicted category-level segmentation
Refined category-level segmentation
Instance-level segmentation
Category-level segmentation: person, bicycle, car.
Instance number prediction person: 4, bicycle: 2.
person
car
person
bicycle
bicycle
person
person
bicycle
bicycle
person
person
person
bicycle
bicycle
Category-level segmentation: dog, cat
Instance number prediction: dog: 2
dog
cat
dog
dog
dog

Fig. 4. The exemplar segmentation results by refining the category-level segmentation with the predicted numbers of instances. For each image, we show their classification results inferred from category-level segmentation and the predicted numbers of instances in the left. In the first row, the refining strategy is to convert the inconsistent predicted labels into background. In the second row, the refining strategy is to convert the wrongly predicted labels in category-level segmentation to the ones predicted in the number vector of instances. Different colors indicate different object instances. Better viewed in zoomed-in color pdf file.

Network Training. To train the whole instance-level network, the over loss function L for each image is actuated as

$$L(\mathbf{t}, \mathbf{t}^*, \mathbf{g}, \mathbf{g}^*) = \lambda L^o(\mathbf{t}, \mathbf{t}^*) + L^n(\mathbf{g}, \mathbf{g}^*). \quad (5)$$

The parameter λ is empirically set to 10, which means the bias towards better pixel-wise instance location prediction. In this way, the number predictions of instances and pixel-wise instance location predictions are jointly optimized in a unified network. The two different targets can benefit each other by learning more robust and discriminative shared convolutional filters. We borrow the convolutional filters except for those of the last prediction layer in the previously trained category-level network to initialize the parameters of the instance-level network. We randomly initialize all newly added layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. The network can be trained by back-propagation and stochastic gradient descent (SGD) [36].

3.3 Testing Stage

During testing, we first feed the input image I into the category-level segmentation network to obtain category-level segmentation results, and then pass the input image into the instance-level network to get the number vector $\mathbf{g}$ of instances and the pixel-wise instance location predictions $\mathbf{t}$.

Then the clustering based on all the predicted instance locations $\mathbf{t}$ of all the pixels can be performed. We separately cluster the predicted instance locations for each category, which can be obtained by filtering the $\mathbf{t}$ with the category-level segmentation result $\mathbf{p}$, and the predicted numbers of instances of all categories $\mathbf{g}$ indicate the expected numbers of clusters used for spectral clustering. The simple normalized spectral clustering [37] is utilized due to its simplicity and effectiveness. For each category c, the similarity matrix W is constructed by calculating the similarities between any pair of pixels that belong to the resulting segmentation mask $\mathbf{p_c}$. Let the spatial coordinate vectors for the pixel i and j be $q_i = [i_x, i_y]$ and $q_j = [j_x, j_y]$, respectively. The t_i and q_i vectors are all normalized by their corresponding maximum. The Gaussian similarity function $w_{i,j}$ for each pair (i, j) is computed as

$$w_{i,j} = \exp\left(\frac{-||t_i - t_j||^2/|t_i|}{2\sigma^2}\right) + \exp\left(\frac{-||q_i - q_j||^2/|q_i|}{2\sigma^2}\right), \quad (6)$$

where $|t_i|$ denotes the feature dimension for the vector t_i, which equals 4 (the coordinates of top-left corner and bottom-right corner), and $|q_i|$ indicates the feature dimension of q_i, which equals 2. During clustering, we simply connect all pixels in the same segmentation mask of a specific category with positive similarity because the local neighboring relationships can be captured by the second term in Eqn. (6). We simply set $\sigma = 0.5$ for all images. To make the clustering results robust to the initialization of seeds during the k-means step of spectral clustering, we randomly select the seeds twenty times by balancing the accuracy and computational cost. Then the clustering result with maximal average within-cluster similarities for all clusters is selected as the final result.

Note that inconsistent global image category predictions from number vectors of instances and pixel-wise category-level segmentation are often observed. For example, as illustrated in the first row of Fig. 4, the number prediction of instances infers 4 person instances and 2 bicycle instances while the category-level segmentation indicates three categories (i.e., person, bicycle, car) appearing in the image. Thus it is necessary to keep the predicted global image category to be consistent between the number prediction of instances and the pixel-wise segmentation. Note that the number prediction task of instances is much simpler than pixel-wise semantic segmentation due to dense pixel-wise optimization targets. We can thus use the number prediction of instances to refine the produced category-level segmentation.

The object category from the number prediction of instances can be easily obtained by thresholding the number vector of instances by $\tau = 0.5$, which means, if the predicted number of instances of a specific category c is larger than τ, the category c is regarded as the true label. Specifically, two strategies are adopted: first, if more than one category is predicted to have at least one instance in the image, any pixels assigned with all other categories will be re-labeled as the background, as illustrated in the first row of Fig. 4; second, if only one category is inferred from the number prediction of instances, pixels labeled with other object

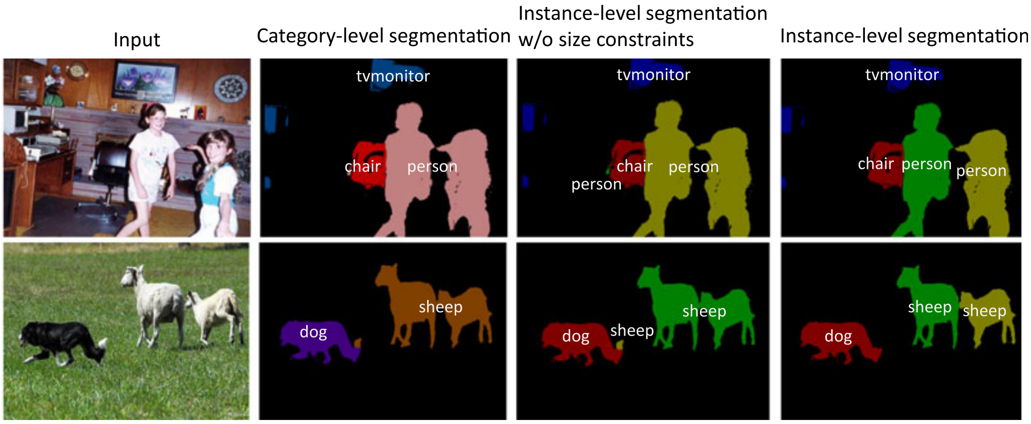


Fig. 5. Comparison of segmentation results by constraining the number of pixels of each clustered object instance. The version without size constraints wrongly clusters two neighboring instances as one instance while our full version can precisely separate them.

categories (excluding background pixels) in the semantic segmentation mask will be totally converted into the predicted ones, as illustrated in the second row of Fig. 4. The refined category-level segmentation masks are used to further generate instance-level segments.

In addition, the predicted segmentation result is not perfect due to the noisy background pixels. The instance locations of pixels belonging to one object have much higher possibilities to form a cluster while the predictions of background pixels are often quite random, forming very small clusters. Therefore, we experimentally discard those clusters, whose numbers of pixels are less than 0.1 percent of the pixels in the segmentation mask. Finally, after obtaining the final clustering result for each category, the instance-level object segmentation result can be easily obtained by combining all the clustering results of all categories. Example results after constraining the number of pixels of each clustered instance region are shown in Fig. 5.

4 Experiments

4.1 Experimental Settings

Dataset and Evaluation Metrics. The proposed PFN is extensively evaluated on the PASCAL VOC 2012 validation segmentation benchmark [38]. We compare our method with four state-of-the-art algorithms: SDS [1], CFM [28], Chen et al. [2] and MNC [27]. Following the baselines [1], [2], [26], the instance-level segmentation annotations from SBD dataset [39] are used when training all variants of the PFN, since VOC 2012 did not provide all annotations of the training set for instance-level segmentation. The training set for the segmentation task on VOC 2012 is used for training all the models and we use the 1,449 images in the validation set for testing. Note that, both SBD dataset [39] and VOC 2012 [38] provide annotations of instance-level object segmentation for the 1,449 images on VOC 2012 validation set. As compared in [2], VOC 2012 provides very elaborated segmentation annotations (e.g., carefully labeled skeletons for a bicycle) for each instance while SBD just gives the whole region (e.g., a rough region for a bicycle). Since Chen et al. [2] re-evaluated the performance of the method in [1] with the annotations from VOC 2012 validation set, most of our evaluations are thus performed with respect to the annotations from VOC 2012 [38] when comparing with [1], [2]. For comparison with HC [26], CFM [28] and MNC [27] that use 5,623 images for training and 5,732 for testing according to VOC 2012 detection split, we also evaluate the performance of PFN with the annotations from SBD dataset [39]. For fair comparison with state-of-the-art instance level segmentation methods, AP^r and AP^r_{vol} metrics are used following SDS [1]. The AP^r metric measures the average precision under 0.5 IoU overlap with ground-truth segmentation. Hariharan et al. [2] proposed to vary IoU scores from 0.1 to 0.9 to show the performance for different applications. The AP^r_{vol} metric calculates the mean of AP^r under all IoU scores. Note that two baselines fine-tune the networks based on Alexnet architecture [40]. For fair comparison, we also report results based on the Alexnet architecture [40]. Our PFN directly calculated the mAP^r between the clustered instance-level object segments with the ground-truth results.

Training Strategies. All networks in our experiments are trained and tested based on the published DeepLab code [18], which is implemented based on the publicly available Caffe platform [41] on a single NVIDIA GeForce Titan GPU with 6 GB memory. We first train the category-level segmentation network, which is then used to initialize our instance-level segmentation network for fine tuning. For both training stages, we randomly initialize all new layers by drawing weights from a zero-mean Gaussian distribution with standard deviation 0.01. We use mini-batch size of 8 images, initial learning rate of 0.001 for pre-trained layers, and 0.01 for newly added layers in all our experiments. We decrease the learning rate to 1/10 of the previous one after 20 epochs and train the two networks for roughly 60 epochs one after the other. The momentum and the weight decay are set as 0.9 and 0.0005, respectively. The same training setting is utilized for all our compared network variants.

We evaluate the testing time by averaging the running time for images on the VOC 2012 validation set on NVIDIA GeForce Titan GPU and Intel Core i7-4930K CPU @3.40GHZ. Our PFN can rapidly process one 300×500 image in about one second, which includes 0.4 second for two network prediction and 0.6 second for clustering on average. This compares much favorably to other state-of-the-art approaches, as the current state-of-the-art methods [1], [2] rely on region proposal pre-processing and complex post-processing

TABLE 1
Comparison of Instance-Level Segmentation Performance with Four State-of-the-Arts Using Mean AP^r Metric over 20 Classes at 0.5 and 0.7 IoU Scores, When Evaluating with the Ground-Truth Annotations from SBD Dataset, Including 5,623 Images for Training and 5,732 for Testing

mAP^r	SDS [1]	HC [26]	CFM [28]	MNC [27]	PFN (ours)
0.5	49.7	60.0	60.7	63.5	**64.4**
0.7	-	40.4	-	41.5	**43.2**

All numbers are in %.

steps: [1] takes about 40 seconds while [2] is excepted to be more expensive than [1] because more complex top-down category specific reasoning and shape prediction are further employed for inference based on [1].

4.2 Results and Comparisons

In the first training stage, we train a category-level segmentation network using the same architecture as "DeepLab-CRF-LargeFOV" in [18]. By evaluating the pixel-wise segmentation in terms of pixel intersection-over-union (IOU) averaged across 21 classes, we achieve 67.53 percent on category-level segmentation task on the PASCAL VOC 2012 validation set [38], which is only slightly inferior to 67.64 percent reported in [18].

Table 1 provides the results of SDS [1], HC [26], CFM [28], MNC [27] and our proposed PFN for instance-level segmentation evaluation with respect to the annotations from SBD dataset [39]. HC [26] used the "hypercolumn" feature representation for each pixel to combine outputs of all units above at all layers of the CNN in order to incorporate rich information. CFM [28] proposed to segment out feature maps via region proposals. MNC [27] sequentially predicts bounding boxes of objects, binary masks and instance segments in a cascaded network architecture. However, different from the feature enhancement in HC [26], feature masking in CFM [28] and cascaded prediction in MNC [27], our PFN directly resolves the pixel grouping of each instance by enforcing that pixels belonging to same instances have coherent representations and predictions of instance locations. The proposed PFN outperforms the previous approaches by a significant margin, averagely 14.7 percent better than SDS [1], 4.4 percent better than HC [26], 3.7 percent than CFM [28] and 0.9 percent than MNC [27] in terms of mean AP^r metric at 0.5 IoU score. When evaluating on 0.7 IoU score, a 2.8 percent improvement in AP^r is obtained by comparing our PFN with HC [26]. We can only compare the results evaluated on 0.5 to 0.7 IoU scores, since no other results evaluated on higher IoU scores are reported for the baselines. Previous methods [28], [26] employed two separated steps for object proposal generation and object recognition, respectively. As a result, the inaccurate proposals may severely degenerate the final segmentation performance. In addition, the separate processing steps obstruct introducing the elaborately crafted neural network tricks (e.g., hypercolumn and convolutional feature masks) for refining object masks. On the contrary, our PFN directly optimizes the final structured outputs that correspond to the target instance-level object segmentation. The intermediate convolutional features in our framework are learned to simultaneously exploit the region grouping of different object instances and the semantic object categorization. The unified joint optimization of our PFN is the key to achieving better segmentation and categorization performance.

Tables 2 and 5 present the comparison of the proposed PFN with two state-of-the-art methods [2], [39] using AP^r metric at IoU score 0.5 and 0.6 to 0.9, respectively. We directly use their published results on PASCAL VOC 2012 validation set for fair comparison. All results of the state-of-the-art methods were reported in [2] which re-evaluated the performance of [39] using VOC 2012 validation set. For fair comparison, we also report the results of PFN using the Alexnet architecture [40] as used in two baselines [2], [39], i.e., "PFN Alexnet". Following the strategy presented in [18], we convert the fully connected layers in Alexnet to fully convolutional layers, and all other settings are the same as those used in "PFN". The results of [39] and [2] achieve 43.8 and 46.3 percent in AP^r metric at IoU 0.5. Meanwhile, our "PFN Alexnet" is significantly superior over the two baselines, i.e., 53.9 percent versus 43.8 percent [39] and 46.3 percent [2] in AP^r metric. Further detailed comparisons in AP^r over 20 classes at IoU scores 0.6 to 0.9 are listed in Table 5. By utilizing the more powerful VGG-16 network architecture, our PFN can substantially improve the performance and outperform these two baselines by over 18.5 percent for SDS [39] and 16.0 percent for the method of Chen et al. [2]. PFN also gives huge boosts in AP^r metrics at 0.6 to 0.9 IoU scores, as reported in Table 5. For example, when evaluating at 0.9 IoU score where the localization accuracy for object instances is strictly required, the two baselines achieve 0.9 percent for SDS [39] and 2.6 percent for [2] while PFN obtains 17.1 percent. This verifies the effectiveness of our PFN although it does not require extra region proposal extractions as the pre-processing step. The detailed AP^r scores for each class are also listed. In

TABLE 2
Comparison of Instance-Level Segmentation Performance with Two State-of-the-Arts Using AP^r Metric over 20 Classes at 0.5 IoU on the PASCAL VOC 2012 Validation Set, Including 9,906 Images for Training and 1,449 Images for Testing

Settings	Method	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	average
Baselines	SDS [39]	58.8	0.5	60.1	34.4	29.5	60.6	40.0	73.6	6.5	52.4	31.7	62.0	49.1	45.6	47.9	22.6	43.5	26.9	66.2	66.1	43.8
	Chen et al. [2]	63.6	0.3	61.5	43.9	33.8	67.3	46.9	74.4	8.6	52.3	31.3	63.5	48.8	47.9	48.3	26.3	40.1	33.5	66.7	67.8	46.3
Ours (Alexnet)	PFN Alexnet	64.7	18.7	71.2	50.1	25.2	65.9	35.8	84.8	14.6	62.6	42.8	76.4	78.8	64.7	38.4	34.8	43.2	60.1	80.2	65.7	53.9
Ours (VGG 16)	PFN	**79.2**	**25.6**	**77.8**	**58.1**	**38.4**	**75.1**	**40.2**	**92.9**	**19.0**	**74.5**	**49.2**	**84.1**	**83.2**	**75.1**	**46.6**	**43.9**	**58.9**	**64.6**	**86.2**	**73.7**	**62.3**

All numbers are in %.

TABLE 3
Comparison of Instance-Level Segmentation Performance with Different Architecture Variants of Our PFN Using AP^r Metric over 20 Classes at 0.5 IoU on the Randomly Selected 1,449 Images from PASCAL VOC 2012 Train Set for the Validation

Settings	Method	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	average
Training Strategy	PFN unified	74.5	22.2	**80.5**	57.3	27.7	70.1	21.2	75.6	17.2	66.5	48.9	72.8	73.1	65.7	43.2	12.5	29.5	49.4	75.3	59.2	52.1
Network structure	PFN w/o multiscale	74.3	17.6	72.9	58.2	28.3	73.2	32.1	**93.4**	18.3	70.3	**49.1**	81.2	79.6	72.2	47.9	25.3	52.0	62.9	88.1	**72.6**	58.5
	PFN fusing_summation	77.3	25.0	75.7	57.1	37.0	73.9	40.1	92.6	18.1	73.2	47.6	82.8	82.7	75.4	44.8	42.4	56.6	63.5	85.7	71.9	61.2
The number of instances	PFN w/o category-level	75.7	21.3	79.5	**58.3**	34.3	65.4	36.1	79.2	16.8	63.5	40.1	72.1	71.2	69.2	46.2	15.2	44.9	48.2	74.3	67.8	54.0
	PFN w/o instance-level	77.1	24.6	75.1	56.9	36.9	73.3	39.8	91.2	17.3	73.5	47.1	83.1	82.8	75.8	44.1	41.4	54.2	64.1	85.1	71.9	60.8
	PFN separate_finetune	77.3	23.1	74.2	57.8	37.1	73.7	39.2	92.1	17.6	73.8	47.5	82.7	82.9	75.3	44.9	42.1	55.8	63.0	85.2	71.7	60.9
Testing strategy	PFN w/o coordinates	72.3	16.2	71.3	56.8	28.6	73.1	30.5	91.9	17.3	64.2	47.9	81.5	80.4	72.8	**48.3**	24.2	52.7	63.2	**87.8**	71.6	57.6
	PFN w/o classify+size	76.0	22.2	73.1	55.9	32.5	73.1	34.6	91.9	17.4	70.1	47.4	80.2	82.4	74.0	43.5	40.5	54.7	62.1	84.2	70.1	59.3
	PFN w/o size	76.2	**25.6**	74.7	57.1	35.2	74.0	39.4	92.3	17.6	73.8	48.3	82.7	83.0	75.4	44.7	41.0	53.8	63.6	85.5	70.6	60.7
	PFN w/o classify	77.3	23.6	74.8	56.9	34.4	73.6	36.5	92.4	17.7	72.3	47.8	81.6	82.5	74.3	44.8	41.8	55.3	63.2	84.0	71.8	60.3
Ours (VGG 16)	PFN	**78.2**	25.3	76.1	57.5	**37.2**	**74.3**	**40.4**	93.2	**18.7**	**74.1**	48.1	**83.2**	**83.2**	**76.2**	45.9	**42.9**	**57.2**	**64.2**	86.0	72.1	**61.7**
Upperbound	PFN upperbound_instnum	84.9	26.1	84.3	62.7	36.8	80.8	36.2	95.1	21.7	85.2	60.8	84.1	86.9	77.4	52.1	24.3	59.1	72.3	93.9	73.8	64.9
	PFN upperbound_instloc	86.6	30.1	88.9	70.2	42.9	78.7	39.1	96.4	28.3	87.1	64.8	86.2	92.8	82.1	57.1	30.9	68.1	75.1	96.5	76.3	68.9
	PFN upperbound_category	92.3	84.1	94.5	86.9	82.5	86.6	85.3	97.1	86.7	84.2	82.3	90.5	95.3	86.5	84.3	86.7	82.3	85.8	98.1	85.1	87.8
Lowerbound	PFN clustering	52.1	0.1	53.1	26.1	22.5	54.8	35.1	60.1	1.2	46.1	26.9	50.1	38.8	37.6	40.3	16.8	37.1	22.5	59.6	58.3	37.0

All numbers are in %.

general, our method shows dramatically higher performance than the baselines. Especially, in predicting small object instances (e.g., bird and chair) or object instances with a lot of occlusion (e.g., table and sofa), our method achieves a larger gain. This demonstrates that our network can effectively deal with the internal boundaries between the object instances and robustly predict the instance-level masks with various appearance patterns or occlusion.

4.3 Ablations Studies of our Networks

We further evaluate the effectiveness of our important components of PFN, including the training strategy, network structure, the number prediction of instances, testing strategy and upperbounds, respectively. To validate different architecture variants of our PFN more fairly, we randomly select 1,449 images from the training set as the validation set for all our ablation experiments and the rest images in the training set for training the networks. The performance over all the categories by all variants is reported in Tables 3 and 4.

Training Strategy. Note that our PFN training includes two stages: the category-level segmentation network and the instance-level network. To justify the necessity of using two stages, we evaluate the performance of training a unified network that consists of the category-level segmentation, pixel-wise instance location prediction and the number prediction of instances in one learning stage, namely "PFN unified". "PFN unified" is fine-tuned based on the VGG-16 pre-trained model and three losses for three sub-tasks are optimized in one network. The category-level prediction is appended in the last convolutional layer within the dashed blue box in Fig. 3, and the loss weight for category-level segmentation is set as 1. From our experimental results, "PFN unified" leads to 9.6 percent decrease in average AP^r and 6.4 percent decrease in average AP^r_{vol}, compared with "PFN". Intuitively, the target of category-level segmentation

TABLE 4
Comparison of Instance-Level Segmentation Performance with Different Architecture Variants of Our PFN Using AP^r_{vol} Metric over 20 Classes That Averages All AP^r Performance from 0.1 to 0.9 IoU Scores on the Randomly Selected 1,449 Images from PASCAL VOC 2012 Train Set for the Validation

Settings	Method	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	average
Training Strategy	PFN unified	71.1	24.2	74.8	47.8	25.9	61.4	21.8	72.2	16.3	60.9	43.3	72.5	65.6	60.2	39.3	17.8	37.9	49.2	68.6	54.1	49.2
Network structure	PFN w/o multiscale	72.3	26.8	66.9	51.3	28.9	**68.8**	28.5	85.2	20.2	62.6	46.7	73.8	67.9	64.2	43.2	**25.4**	45.1	56.9	76.8	59.1	53.5
	PFN fusing_summation	73.5	28.8	74.9	49.7	30.2	65.8	28.9	**85.4**	**23.8**	**66.4**	49.8	78.9	**73.2**	66.4	42.4	25.2	**51.0**	57.9	**79.1**	59.2	55.5
Instance number	PFN w/o category-level	72.3	26.1	74.6	48.9	29.1	62.1	22.0	73.6	17.9	61.8	48.6	73.9	66.3	61.6	40.4	18.2	38.1	50.8	69.3	55.6	50.6
	PFN w/o instance-level	72.4	27.1	73.9	49.1	30.1	64.8	28.3	84.9	22.6	66.2	49.3	78.8	72.2	65.8	42.3	24.6	50.8	56.3	78.6	58.7	54.8
	PFN separate_finetune	72.3	27.2	73.8	48.1	30.2	66.0	**29.7**	84.1	22.4	65.8	48.5	78.2	71.9	65.5	42.7	24.3	50.1	56.8	78.1	58.2	54.7
Testing strategy	PFN w/o coordinates	70.8	26.0	70.4	49.2	28.1	65.2	27.6	83.2	18.5	62.2	46.8	75.2	69.4	64.7	43.2	23.8	46.4	56.2	77.8	59.4	53.2
	PFN w/o classify+size	71.4	28.2	71.5	48.5	27.1	65.4	28.2	83.6	20.1	63.6	48.1	76.8	69.8	63.6	42.1	24.1	47.3	57.1	78.0	59.6	53.7
	PFN w/o size	73.5	29.3	**75.8**	50.0	30.8	65.9	28.9	84.6	22.8	65.1	49.2	78.3	71.2	**66.5**	42.7	23.6	50.4	57.9	78.1	60.1	55.2
	PFN w/o classify	73.0	28.9	73.1	49.0	28.6	65.5	28.4	84.1	21.8	64.7	49.2	77.9	70.4	64.1	42.7	23.8	49.1	57.6	78.2	59.9	54.5
Ours (VGG 16)	PFN	**73.8**	**29.7**	75.3	**50.1**	**31.1**	66.7	29.2	84.9	23.0	65.8	**50.2**	**79.3**	72.1	65.1	**43.6**	24.4	50.2	**58.2**	78.3	**60.4**	**55.6**

All numbers are in %.

TABLE 5
Per-Class Instance-Level Segmentation Results Using AP^r Metric over 20 Classes at 0.6 to 0.9 (with a Step Size of 0.1) IoU Scores on the VOC PASCAL 2012 Validation Set

IoU score	Method	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motor	person	plant	sheep	sofa	train	tv	average
0.6	SDS [39]	43.6	0	52.8	19.5	25.7	53.2	33.1	58.1	3.7	43.8	29.8	43.5	30.7	29.3	31.8	17.5	31.4	21.2	57.7	62.7	34.5
	Chen et al. [2]	57.1	0.1	52.7	24.9	27.8	62.0	**36.0**	66.8	6.4	45.5	23.3	55.3	33.8	35.8	35.6	20.1	35.2	28.3	59.0	57.6	38.2
	PFN Alexnet	62.5	10.1	69.3	33.4	20.8	55.3	21.9	78.1	10.7	57.2	38.4	73.6	66.4	67.8	31.7	18.1	34.1	51.2	68.6	51.8	46.1
	PFN	**76.1**	**13.8**	**72.5**	**43.3**	**29.1**	**68.8**	29.1	**85.9**	**13.4**	**68.2**	**45.4**	**80.1**	**71.6**	**75.2**	**41.7**	**21.3**	**48.5**	**54.2**	**79.4**	**64.2**	**54.1**
0.7	SDS [39]	17.8	0	32.5	7.2	19.2	47.7	22.8	42.3	1.7	18.9	16.9	20.6	14.4	12.0	15.7	5.0	23.7	15.2	40.5	51.4	21.3
	Chen et al. [2]	40.8	0.07	40.1	16.2	19.6	56.2	26.5	46.1	2.6	25.2	16.4	36.0	22.1	20.0	22.6	7.7	27.5	19.5	47.7	46.7	27.0
	PFN Alexnet	59.8	8.2	61.7	29.4	18.3	52.8	21.9	71.8	6.9	45.4	28.5	66.1	61.2	57.8	24.3	10.7	28.8	41.3	61.2	45.6	40.1
	PFN	**71.0**	**9.6**	**64.3**	**35.7**	**21.8**	**62.9**	**28.9**	**80.4**	**9.3**	**52.4**	**29.7**	**70.7**	**62.8**	**62.5**	**28.4**	**13.3**	**37.5**	**45.8**	**73.6**	**48.1**	**45.4**
0.8	SDS [39]	2.1	0	8.3	4.5	11.5	32.3	9.0	17.9	0.7	4.7	9.0	6.5	1.8	4.4	3.3	1.9	7.9	10.2	12.7	24.3	8.7
	Chen et al. [2]	10.5	0	15.7	9.8	11.4	32.7	12.5	34.8	1.1	11.6	9.5	15.3	4.6	6.5	6.0	3.0	13.9	14.4	27.0	30.4	13.5
	PFN Alexnet	47.9	2.6	49.3	17.2	9.5	45.8	12.1	59.4	3.8	38.2	16.8	48.9	42.1	35.2	15.3	5.2	21.6	25.1	51.8	31.1	28.9
	PFN	**57.3**	**5.8**	**51.3**	**24.2**	**14.3**	**51.8**	**18.1**	**65.2**	**6.7**	**40.1**	**22.1**	**53.9**	**43.4**	**38.2**	**20.3**	**8.5**	**28.1**	**27.4**	**61.2**	**39.2**	**33.9**
0.9	SDS [39]	0	0	0.2	0.3	2.0	3.8	0.2	0.9	0.1	0.2	1.5	0	0	0	0.1	0.1	0	2.3	0.2	5.8	0.9
	Chen et al. [2]	0.6	0	0.6	0.5	4.9	9.8	1.1	8.3	0.1	1.1	1.2	1.7	0.3	0.8	0.6	0.3	0.8	7.6	4.3	6.2	2.6
	PFN Alexnet	38.8	2.2	25.2	8.4	7.2	31.9	6.3	41.7	2.2	24.8	4.9	29.8	15.4	8.2	7.1	3.4	11.5	13.6	24.7	9.2	15.8
	PFN	**44.9**	**3.2**	**26.1**	**8.8**	**8.2**	**32.9**	**7.1**	**43.4**	**2.9**	**26.1**	**5.3**	**30.5**	**16.2**	**9.2**	**7.8**	**3.9**	**15.3**	**15.4**	**25.2**	**10.2**	**17.1**

All numbers are in %.

is to be robust for individual object instances of the same category while erasing the instance-level information during optimization. On the contrary, the instance-level network aims to learn the instance-level information for distinguishing different object instances with large variance in appearance, view or scale. This comparison result verifies well that training with two sequential stages can lead to better global instance-level segmentation.

Network Structure. Extensive evaluations on different network structures are also performed. First, the effectiveness of multi-scale prediction is verified. "PFN w/o multiscale" shows the performance of using only one straightforward layer to predict pixel-wise instance locations. The performance decreases by 3.2 percent in AP^r compared with "PFN". This significant inferiority demonstrates the effectiveness of multi-scale fusing that incorporates the local fine details and global semantic information into predicting the pixel-wise instance locations.

In the fusing layer for predicting pixel-wise instance locations, "PFN" utilizes the concatenation operation instead of element-wise summation for multi-scale prediction. "PFN fusing_summation" shows 0.5 percent decrease in AP^r when compared to "PFN". The 1×1 convolutional filters are utilized to adaptively weigh the contribution of the instance location prediction of each scale, which is more reasonable and experimentally effective than simple summation.

The Number Prediction of Instances. We explore other options to predict the numbers of instances of all categories for each image. "PFN w/o category-level" only utilizes the instance location predictions as the feature maps for predicting numbers of instances and the category-level information is totally ignored. The large gap between "PFN w/o category-level" and "PFN" (54.0 percent versus 61.7 percent) verifies the importance of using category-level information for predicting numbers of instances. Because the instance location predictions only capture the numbers of instances of all categories and category-level information is discarded, the exact number of instances for a specific category thus cannot be inferred. The importance of incorporating instance-level information is also verified by comparing "PFN w/o instance-level" with "PFN", 60.8 percent versus 61.7 percent in AP^r. This shows that the number prediction of instances can benefit from the pixel-wise instance location prediction, where more fine-grained annotations (pixel-wise instance-level locations) are provided for learning better feature maps.

We also evaluate the performance of sequentially optimizing the instance locations and the number of instances instead of using one unified network. "PFN separate_finetune" first optimizes the network for predicting pixel-wise instance locations, and then fixes the current network parameters and only trains the newly added parameters for the number prediction of instances. The performance decrease of "PFN separate_finetune" compared to "PFN" (60.9 percent versus 61.7 percent in AP^r) shows well the effectiveness of training one unified network. The information in the global aspect from numbers of instances can be utilized for predicting more accurate instance locations.

Finally, we report the accuracy of the number prediction of instances by our PFN by calculating the mean absolute errors between the predicted number of instances and the groundtruth of each category for all images. It is observed that the PFN achieves 2.31 numbers in terms of mean absolute errors on average for all categories. The number prediction of instances may fail for the images with many small and occluded instances and thus is a bottleneck for the final performance of our PFN. By refining the category-level segmentation results with the predicted numbers of instances, the category-level segmentation performance in terms of pixel-wise IoU accuracy can be boosted from 67.53 to 68.45 percent.

Testing Strategy. We also test different strategies for generating final instance-level segmentations during testing. Note that during spectral clustering, the similarity of two pixels is computed by considering both the prediction instance locations with four dimensions and two spatial coordinates of each pixel. By eliminating the coordinates in the similarity function, a significant decrease in AP^r can be seen by comparing "PFN w/o coordinates" with "PFN", 57.6 percent versus 61.7 percent. This verifies that the spatial

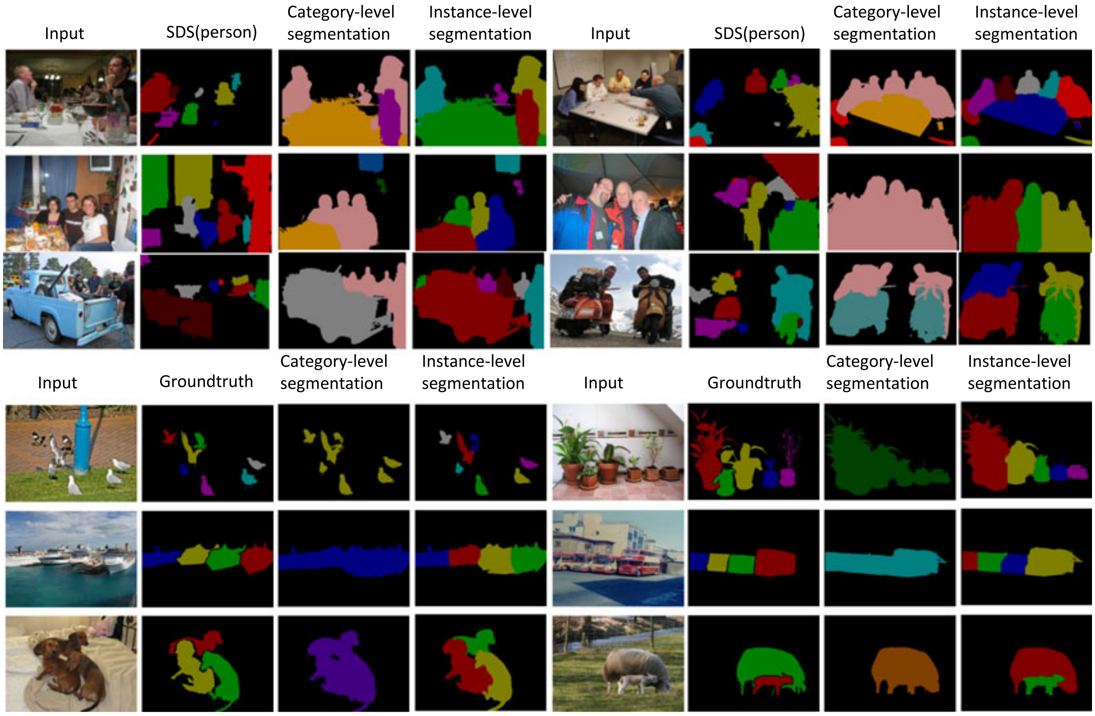

Input
SDS(person)
Category-level segmentation
Instance-level segmentation
Input
SDS(person)
Category-level segmentation
Instance-level segmentation
Input
Groundtruth
Category-level segmentation
Instance-level segmentation
Input
Groundtruth
Category-level segmentation
Instance-level segmentation

Fig. 6. Illustration of instance-level object segmentation results by the proposed PFN. We also compare our results with that of SDS [39]. Since SDS [39] predicts the scores for each proposal of each category, the top 10 predictions for each image of the person category are shown for the visual comparison. Note that for instance-level segmentation results, different colors only indicate different object instances and do not represent the semantic categories. In terms of category-level segmentation, different colors are used to denote different semantic labels. Best viewed in color.

coordinates can enhance the local neighboring connections during clustering, which can lead to more reasonable and meaningful instance-level segmentation results.

In addition, we also test the performance influenced by using different feature representations (i.e., how to use the predicted coordinates) considered in the similarity function during clustering. First, we evaluate the version that uses the predicted center coordinates transformed from the predicted top-left and down-right coordinates as the additional features in the similarity function. It achieves 0.9 percent improvement in terms of AP^r metric on 0.5 IoU over our "PFN" that only uses the predicted top-left and down-right coordinates as features. Second, our PFN achieves 1.3 percent higher performance over the version that uses the predicted center coordinates, height and width as the feature instead of predicted top-left and down-right coordinates. These experiments prove that the redundant feature representation can improve the discriminative capability of segmenting instances during the clustering step.

Recall that two steps are used for post-processing, including refining the segmentation results with the number prediction of instances and constraining the cluster size during clustering. We extensively evaluate the effectiveness of using these two steps. By comparing 59.3 percent of "PFN w/o classify + size" that eliminates these two steps with 61.7 percent of "PFN" in AP^r, better performance can be obtained by leveraging the number prediction of instances and constraining the cluster size to refine instance-level segmentation. Only eliminating the refining strategy by constraining the cluster size results in 1.0 percent decrease. It demonstrates that constraining the cluster size can help reduce the effect of noisy background pixels to some extent and more robust instance level segmentation results can be obtained. On the other hand, the incorporation of the number prediction of instances can help improve the performance in AP^r by 1.4 percent when comparing "PFN w/o classify" with "PFN". In particular, significant improvements are obtained for easily confused categories such as "cow", "sheep" and "horse". This demonstrates the effectiveness of using the number prediction of instances for refining the pixel-wise segmentation results.

Upperbound. We also evaluate the limitations of our current algorithm. First, "PFN upperbound_instnum" reports the performance of using the ground-truth number prediction of instances for clustering and other experimental settings are kept the same. It can be seen that only 3.2 percent improvement in AP^r is obtained. The errors from the number prediction of instances are already small and have only little effect on the final instance-level segmentation. Second, the upperbound for instance location predictions is reported in "PFN upperbound_instloc" by using the ground-truth instance locations for each pixel as the features for clustering. The large gap between 68.9 percent of "PFN upperbound_instloc" and 61.7 percent of "PFN" verifies that the accurate instance location prediction is critical for good instance-level segmentation. Note that the current category-level segmentation only achieves 67.53 percent of pixel-wise IoU score, which largely limits the performance of our instance-level segmentation because we perform the clustering on the category-level segmentation. A better category-level segmentation network architecture can definitely help improve the performance of instance-level segmentation under our PFN framework.

To validate the effectiveness of our designed instance-level network, we also report the upperbound performance of using ground-truth category-level segmentation, the

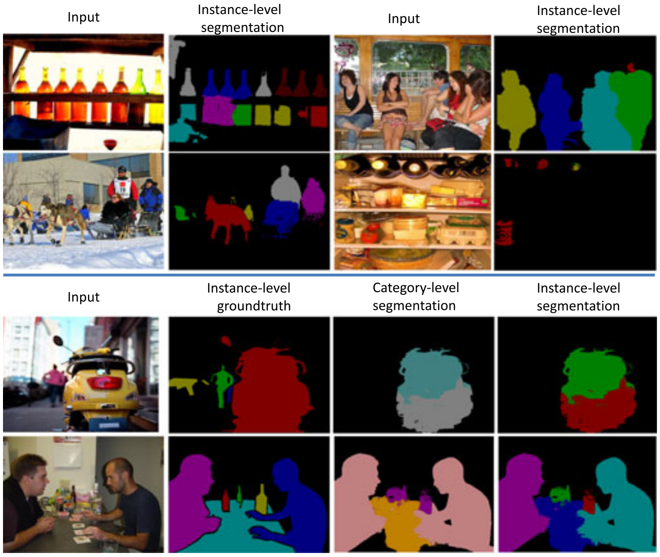

Input
Instance-level segmentation
Input
Instance-level segmentation
Input
Instance-level groundtruth
Category-level segmentation
Instance-level segmentation

Fig. 7. Illustration of failure cases. Our PFN may fail to segment object instances with heavy occlusion (in first row) and small size (in second row). The instance-level segmentation of our PFN is also limited by the accuracy of category-level segmentation prediction (in third and fourth row).

predicted instance locations and numbers to generate the final instance-level segmentation results, i.e., “PFN upperbound_category”. The “PFN upperbound_category” achieves 87.8 percent, much higher than 68.9 percent of “PFN upperbound_instloc”, which verifies the capability of our instance-level segmentation network on predicting good pixel-wise instance locations.

Lowerbound. Finally, we also evaluate the performance of the version that directly clusters the pixels of the same category by using their pixel coordinates and number of instances based on the category-level segmentation results. It obtains 37.0 percent in terms of AP^r metric at 0.5 IoU, which is much worse than ours 61.7 percent. Directly clustering the connected components of category-level segmentation results can only handle the sole instances in the image and fail to detect the adjacent and occluded instances. This experiment demonstrates the effectiveness of predicting the pixel-wise instance locations proposed by our PFN.

4.4 Visual Illustration

Fig. 6 visualizes the predicted instance-level segmentation results with our PFN. We visually compare with SDS [1] and the top 10 detection results for the person category for each image by SDS [1] are visualized. Our method can directly produce exact region segments for each object instance just like the results of category-level segmentation. Different colors indicate different object instances for the instance-level segmentation results. The semantic labels of our instance-level segmentation results are exactly the same with the ones labeled in category-level segmentation results. The proposed PFN performs better in predicting the object instances with heavy occlusion, large background clutters and complex scenes. For example, the first three rows show the results on images with very complex background clutters, several objects with heavy occlusion and diverse appearance patterns. The fourth row illustrates some images with very small object instances, such as birds and potted-plants. The fifth row shows examples where the object instances in one image have high similarity in appearance and much occlusion with each other. Other results show more examples of instance-level images under diverse scenarios and with very challenging poses, scales, views and occlusion. These visualization results further demonstrate the effectiveness of the proposed PFN.

We also show some failure cases of our PFN in Fig. 7. The heavily occluded instances and some small object instances are difficult to identify and segment due to the imprecise prediction for instance location. In addition, the instance-level object segmentation results are also affected by imprecise category-level segmentation.

5 Conclusion and Future Work

In this paper, we present an effective proposal-free network for fine-grained instance-level segmentation. Instead of utilizing extra region proposal methods as the requisite, PFN directly predicts the instance location vector for each pixel that belongs to a specific instance and the numbers of instances of all categories. The pixels that predict the same or close instance locations can be directly regarded as belonging to the same object instance. Significant improvements over the state-of-the-art methods are achieved by PFN on the PASCAL VOC 2012 segmentation benchmark. Extensive evaluations of different components of PFN are conducted to validate the effectiveness of our method. In future work, we plan to extend our framework to the generic multiple instances in outdoor and indoor scenes, which may have higher degrees of clutters and occlusion. Furthermore, the current framework may fail to detect heavily occluded instances due to the wrongly predicted instance locations and numbers of instances. We will address this problem by incorporating the spatial layout contexts of occluded instances into the neural network prediction.

Acknowledgments

This work was supported by State Key Development Program under Grant 2016YFB1001004, the National Natural Science Foundation of China under Grant 61622214, the Science and Technology Planning Project of Guangdong Province under Grant 2017A020208041, Special Program of the NSFC-Guangdong Joint Fund for Applied Research on Super Computation (the second phase), and Guangdong Natural Science Foundation Project for Research Teams under Grant 2017A030312006.

References

[1] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, “Simultaneous detection and segmentation,” in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 297–312.

[2] Y.-T. Chen, X. Liu, and M.-H. Yang, “Multi-instance object segmentation with occlusion handling,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3470–3478.

[3] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun, “Monocular object instance segmentation and depth ordering with CNNs,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 2614–2622.

[4] J. Dai, K. He, and J. Sun, “BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1635–1643.

[5] S. Zheng, et al., “Conditional random fields as recurrent neural networks,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1529–1537.

[6] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.

[8] R. Stewart and M. Andriluka, "End-to-end people detection in crowded scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2325–2333.

[9] C. Szegedy, et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[10] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.

[11] J. Pont-Tuset, P. Arbeláez, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping for image segmentation and object proposal generation," in *arXiv:1503.00848*, Mar. 2015.

[12] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, 2013.

[13] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 391–405.

[14] P. O. Pinheiro, R. Collobert, and P. Dollár, "Learning to segment object candidates," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 1990–1998.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[16] Y. Wei, et al., "HCP: A flexible CNN framework for multi-label image classification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1901–1907, Sep. 2016.

[17] S. Ren, K. He, R. Girshick, X. Zhang, and J. Sun, "Object detection networks on convolutional feature maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1476–1481, Jul. 2017.

[18] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille, "Weakly- and semi-supervised learning of a DCNN for semantic image segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1742–1750.

[19] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1520–1528.

[20] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.

[21] S. Gidaris and N. Komodakis, "Object detection via a multi-region & semantic segmentation-aware CNN model," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1134–1142.

[22] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2155–2162.

[23] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[24] N. Silberman, D. Sontag, and R. Fergus, "Instance segmentation of indoor scenes using a coverage loss," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 616–631.

[25] J. Tighe, M. Niethammer, and S. Lazebnik, "Scene parsing with object instances and occlusion ordering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 3748–3755.

[26] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 447–456.

[27] J. Dai, K. He, and J. Sun, "Instance-aware semantic segmentation via multi-task network cascades," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3150–3158.

[28] J. Dai, K. He, and J. Sun, "Convolutional feature masking for joint object and stuff segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3992–4000.

[29] M. Ren and R. S. Zemel, "End-to-end instance segmentation with recurrent attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 293–301.

[30] Y. Yang, S. Hallman, D. Ramanan, and C. C. Fowlkes, "Layered object models for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1731–1743, Sep. 2012.

[31] J. Uhrig, M. Cordts, U. Franke, and T. Brox, "Pixel-level encoding and depth layering for instance-level semantic labeling," in *Proc. German Conf. Pattern Recognit.*, 2016, pp. 14–25.

[32] M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2858–2866.

[33] A. Kirillov, E. Levinkov, B. Andres, B. Savchynskyy, and C. Rother, "InstanceCut: From edges to instances with multicut," arXiv:1611.08272, 2016.

[34] S. Liu, J. Jia, S. Fidler, and R. Urtasun, "SGN: Sequential grouping networks for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3496–3504.

[35] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," in *Proc. Int. Conf. Learn. Representations*, 2015.

[36] Y. LeCun, et al., "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

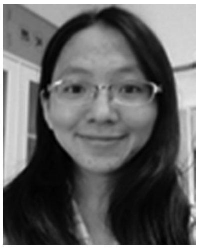
[37] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2002, pp. 849–856.

[38] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, no. 1, pp. 98–136, 2014.

[39] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 991–998.

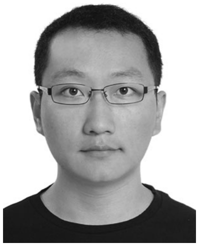
[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[41] Y. Jia, et al., "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

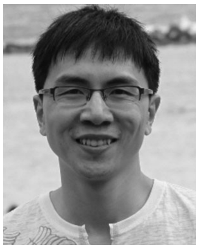


Xiaodan Liang is working toward the PhD degree from Sun Yat-sen University, China, advised by Prof. Liang Lin. She received her BS degree in Software Engineering from Sun Yat-sen University in 2011. Her research interests mainly include semantic segmentation, object/action recognition, and medical image analysis.

Liang Lin is the Executive R&D Director of SenseTime Group Limited and a full Professor of Sun Yat-sen University. He is the Excellent Young Scientist of the National Natural Science Foundation of China. From 2008 to 2010, he was a Post-Doctoral Fellow at University of California, Los Angeles. From 2014 to 2015, as a senior visiting scholar he was with The Hong Kong Polytechnic University and The Chinese University of Hong Kong. He currently leads the SenseTime R&D teams to develop cutting-edges and deliverable solutions on computer vision, data analysis and mining, and intelligent robotic systems. He has authorized and co-authorized on more than 100 papers in top-tier academic journals and conferences (e.g., 12 papers in TPAMI/IJCV and 50+ papers in CVPR/ICCV/NIPS/IJCAI). He has been serving as an associate editor of IEEE Trans. Human-Machine Systems, The Visual Computer and Neurocomputing. He served as Area/Session Chairs for numerous conferences such as ICME, ACCV, ICMR. He was the recipient of Best Paper Dimond Award in IEEE ICME 2017, Best Paper Runners-Up Award in ACM NPAR 2010, Google Faculty Award in 2012, Best Student Paper Award in IEEE ICME 2014, and Hong Kong Scholars Award in 2014. He is a Fellow of IET.

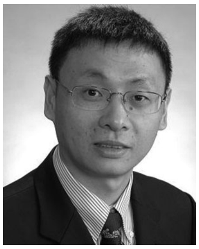


Yunchao Wei is working toward the PhD degree from the Institute of Information Science, Beijing Jiaotong University, China. He is currently working at National University of Singapore as a research intern. His research interests mainly include object classification in computer vision and multi-modal analysis in multimedia.



Xiaohui Shen received the BS and MS degrees from the Department of Automation, Tsinghua University, China, and the PhD degree from the Department of EECS, Northwestern University, in 2013. He is currently a research scientist at Adobe Research, San Jose, California. His research interests include image/video processing and computer vision.

Jianchao Yang (S08, M12) received the MS and PhD degrees in electrical and computer engineering from the University of Illinois at Urbana-Champaign, Urbana, in 2011. His research interests include object recognition, deep learning, sparse coding, image/video enhancement, and deblurring. He is a member of the IEEE.



Shuicheng Yan (M'06-SM'09) is the Vice President and Chief Scientist of Qihoo 360 Technology Co. Ltd., as well as Head of 360 Artificial Intelligence Institute. He is also a tenured Associate Professor at National University of Singapore, and IEEE Fellow, IAPR Fellow and ACM Distinguished Scientist. His research areas include computer vision, machine learning and multimedia analysis, and he has authored/co-authored about 500 high quality technical papers, with Google Scholar citation over 25,000 times and H-index 70. He is TR Highly Cited Researcher of 2014, 2015 and 2016. Besides, his team received 7 times winner or honorable-mention prizes in 5 years over PASCAL VOC and ILSVRC competitions which are core competitions in the field of computer vision, along with more than 10 times best (student) paper awards and especially a Grand Slam in ACM MM, the top conference in the field of multimedia, including Best Paper Award, Best Student Paper Award and Best Demo Award.