Efficient Piecewise Training of Deep Structured Models for Semantic Segmentation

Guosheng Lin, Chunhua Shen, Anton van den Hengel, Ian Reid
The University of Adelaide; and Australian Centre for Robotic Vision

Abstract

Recent advances in semantic image segmentation have mostly been achieved by training deep convolutional neural networks (CNNs). We show how to improve semantic segmentation through the use of contextual information; specifically, we explore 'patch-patch' context between image regions, and 'patch-background' context. For learning from the patch-patch context, we formulate Conditional Random Fields (CRFs) with CNN-based pairwise potential functions to capture semantic correlations between neighboring patches. Efficient piecewise training of the proposed deep structured model is then applied to avoid repeated expensive CRF inference for back propagation. For capturing the patch-background context, we show that a network design with traditional multi-scale image input and sliding pyramid pooling is effective for improving performance. Our experimental results set new state-of-the-art performance on a number of popular semantic segmentation datasets, including NYUDv2, PASCAL VOC 2012, PASCAL-Context, and SIFT-flow. In particular, we achieve an intersection-over-union score of 78.0 *on the challenging PASCAL VOC 2012 dataset.*

1. Introduction

Semantic image segmentation aims to predict a category label for every image pixel, which is an important yet challenging task for image understanding. Recent approaches have applied convolutional neural network (CNNs) [13, 32, 3] to this pixel-level labeling task and achieved remarkable success. Among these CNN-based methods, fully convolutional neural networks (FCNNs) [32, 3] have become a popular choice, because of their computational efficiency for dense prediction and end-to-end style learning.

Contextual relationships are ubiquitous and provide important cues for scene understanding tasks. Spatial context can be formulated in terms of semantic compatibility relations between one object and its neighboring objects or image patches (stuff), in which a compatibility relation is an indication of the co-occurrence of visual patterns. For example, a car is likely to appear over a road, and a glass is likely to appear over a table. Context can also encode incompatibility relations. For example, a car is not likely to be surrounded by sky. These relations also exist at finer scales, for example, in object part-to-part relations, and part-to-object relations. In some cases, contextual information is the most important cue, particularly when a single object shows significant visual ambiguities. A more detailed discussion of the value of spatial context can be found in [21].

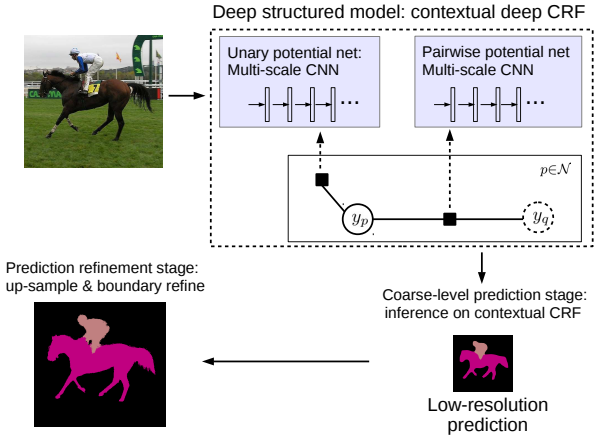


Figure 1. An illustration of the prediction process of our method. Both our unary and pairwise potentials are formulated as multi-scale CNNs for capturing semantic relations between image regions. Our method outputs low-resolution prediction after CRF inference, then the prediction is up-sampled and refined in a standard post-processing stage to output the final prediction.

We explore two types of spatial context to improve the segmentation performance: patch-patch context and patch-background context. The patch-patch context is the semantic relation between the visual patterns of two image patches. Likewise, patch-background context is the semantic relation between a patch and a large background region.

Explicitly modeling the patch-patch contextual relations has not been well studied in recent CNN-based segmentation methods. In this work, we propose to explicitly model the contextual relations using conditional random fields (CRFs). We formulate CNN-based pairwise potential functions to capture semantic correlations between neighboring

patches. Some recent methods combine CNNs and CRFs for semantic segmentation, e.g., the dense CRFs applied in [3, 40, 48, 5]. The purpose of applying the dense CRFs in these methods is to refine the upsampled low-resolution prediction to sharpen object/region boundaries. These methods consider Potts-model-based pairwise potentials for enforcing local smoothness. There the pairwise potentials are conventional log-linear functions. In contrast, we learn more general pairwise potentials using CNNs to model the semantic compatibility between image regions. Our CNN pairwise potentials aim to improve the coarse-level prediction rather than doing local smoothness, and thus have a different purpose compared to Potts-model-based pairwise potentials. Since these two types of potentials have different effects, they can be combined to improve the segmentation system. Fig. 1 illustrates our prediction process.

In contrast to patch-patch context, patch-background context is widely explored in the literature. For CNN-based methods, background information can be effectively captured by combining features from a multi-scale image network input, and has shown good performance in some recent segmentation methods [13, 33]. A special case of capturing patch-background context is considering the whole image as the background region and incorporating the image-level label information into learning. In our approach, to encode rich background information, we construct multi-scale networks and apply sliding pyramid pooling on feature maps. The traditional pyramid pooling (in a sliding manner) on the feature map is able to capture information from background regions of different sizes.

Incorporating general pairwise (or high-order) potentials usually involves expensive inference, which brings challenges for CRF learning. To facilitate efficient learning we apply piecewise training of the CRF [43] to avoid repeated inference during back propagation training.

Thus our main *contributions* are as follows.

1. We formulate CNN-based general pairwise potential functions in CRFs to explicitly model patch-patch semantic relations.

2. Deep CNN-based general pairwise potentials are challenging for efficient CNN-CRF joint learning. We perform approximate training, using piecewise training of CRFs [43], to avoid the repeated inference at every stochastic gradient descent iteration and thus achieve efficient learning.

3. We explore background context by applying a network architecture with traditional multi-scale image input [13] and sliding pyramid pooling [26]. We empirically demonstrate the effectiveness of this network architecture for semantic segmentation.

4. We set new state-of-the-art performance on a number of popular semantic segmentation datasets, including NYUDv2, PASCAL VOC 2012, PASCAL-Context, and SIFT-flow. In particular, we achieve an intersection-over-union score of 78.0 on the PASCAL VOC 2012 dataset, which is *the best reported result* to date.

1.1. Related work

Exploiting contextual information has been widely studied in the literature (e.g., [39, 21, 7]). For example, the early work "TAS" [21] models different types of spatial context between *Things* and *Stuff* using a generative probabilistic graphical model.

The most successful recent methods for semantic image segmentation are based on CNNs. A number of these CNN-based methods for segmentation are region-proposal-based methods [14, 19], which first generate region proposals and then assign category labels to each. Very recently, FCNNs [32, 3, 5] have become a popular choice for semantic segmentation, because of their effective feature generation and end-to-end training. FCNNs have also been applied to a range of other dense-prediction tasks recently, such as image restoration [10], image super-resolution [8] and depth estimation [11, 29]. The method we propose here is similarly built upon fully convolution-style networks.

The direct prediction of FCNN based methods usually are in low-resolution. To obtain high-resolution predictions, a number of recent methods focus on refining the low-resolution prediction to obtain high resolution prediction. DeepLab-CRF [3] performs bilinear upsampling of the prediction score map to the input image size and apply the dense CRF method [24] to refine the object boundary by leveraging the color contrast information. CRF-RNN [48] extends this approach by implementing recurrent layers for end-to-end learning of the dense CRF and the FCNN network. The work in [35] learns deconvolution layers to upsample the low-resolution predictions. The depth estimation method [30] explores super-pixel pooling for building the gap between low-resolution feature map and high-resolution final prediction. Eigen *et al.* [9] perform coarse-to-fine learning of multiple networks with different resolution outputs for refining the coarse prediction. The methods in [18, 32] explore middle layer features (skip connections) for high-resolution prediction. Unlike these methods, our method focuses on improving the coarse (low-resolution) prediction by learning general CNN pairwise potentials to capture semantic relations between patches. These refinement methods are complementary to our method.

Combining the strengths of CNNs and CRFs for segmentation has been the focus of several recently developed approaches. DeepLab-CRF in [3] trains FCNNs and applies a dense CRF [24] method as a post-processing step. CRF-RNN [48] and the method in [40] extend DeepLab and [25] by jointly learning the dense CRFs and CNNs. They consider Potts-model based pairwise potential functions which enforce smoothness only. The CRF model in these methods is for refining the up-sampled prediction. Unlike these methods, our approach learns CNN-based pairwise potential functions for modeling semantic

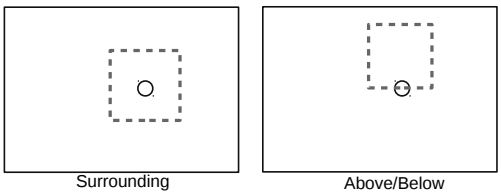


Figure 2. An illustration of constructing pairwise connections in a CRF graph. A node is connected to all other nodes which lie within the range box (dashed box in the figure). Two types of spatial relations are described in the figure, which correspond to two types of pairwise potential functions.

relations between patches.

Jointly learning CNNs and CRFs has also been explored in other applications apart from segmentation. The recent work in [29, 30] proposes to jointly learn *continuous* CRFs and CNNs for depth estimation from single monocular images. The work in [45] combines CRFs and CNNs for human pose estimation. The authors of [4] explore joint training of Markov random fields and deep neural networks for predicting words from noisy images and image s classification. Different from these methods, we explore efficient piecewise training of CRFs with CNN pairwise potentials.

2. Modeling semantic pairwise relations

Fig. 3 conceptualizes our architecture at a high level. Given an image, we first apply a convolutional network to generate a feature map. We refer to this network as 'FeatMap-Net'. The resulting feature map is at a lower resolution than the original image because of the down-sampling operations in the pooling layers.

We then create the CRF graph as follows: for each location in the feature map (which corresponds to a rectangular region in the input image) we create one node in the CRF graph. Pairwise connections in the CRF graph are constructed by connecting one node to all other nodes which lie within a spatial range box (the dashed box in Fig. 2). We consider different spatial relations by defining different types of range box, and each type of spatial relation is modeled by a specific pairwise potential function. As shown in Fig. 2, our method models the "surrounding" and "above/below" spatial relations. In our experiments, the size of the range box (dash box in the figure) size is $0.4a \times 0.4a$. Here we denote by a the length of the short edge of the feature map.

Note that although 'FeatMap-Net' defines a common architecture, in fact we train three such networks: one for the unary potential and one each for the two types of pairwise potential.

3. Contextual Deep CRFs

Here we describe the details of our deep CRF model. We denote by $\boldsymbol{x} \in \mathcal{X}$ one input image and $\boldsymbol{y} \in \mathcal{Y}$ the labeling mask which describes the label configuration of

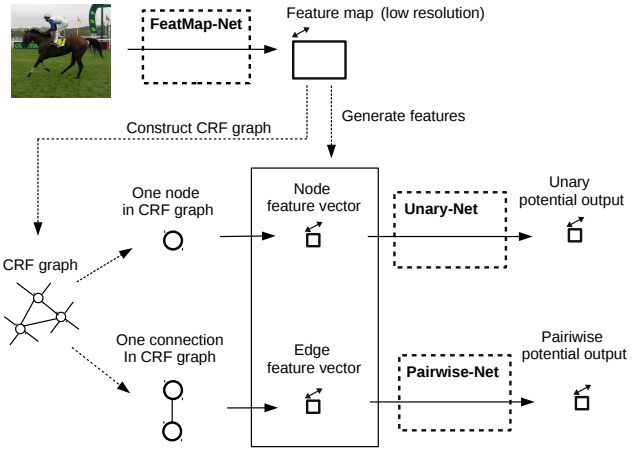


Figure 3. An illustration of generating unary or pairwise potential function outputs. First a feature map is generated by a FeatMap-Net, and a CRF graph is constructed based on the spatial resolution of the feature map. Finally the Unary-Net (or Pairwise-Net) produces potential function outputs.

each node in the CRF graph. The energy function is denoted by $E(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{\theta})$ which models the compatibility of the input-output pair, with a small output value indicating high confidence in the prediction $\boldsymbol{y}$. All network parameters are denoted by $\boldsymbol{\theta}$ which we need to learn. The conditional likelihood for one image is formulated as follows:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp[-E(\boldsymbol{y}, \boldsymbol{x})]. \quad (1)$$

Here $Z(\boldsymbol{x}) = \sum_{\boldsymbol{y}} \exp[-E(\boldsymbol{y}, \boldsymbol{x})]$ is the partition function. The energy function is typically formulated by a set of unary and pairwise potentials:

$$E(\boldsymbol{y}, \boldsymbol{x}) = \sum_{U \in \mathcal{U}} \sum_{p \in \mathcal{N}_U} U(y_p, \boldsymbol{x}_p) + \sum_{V \in \mathcal{V}} \sum_{(p,q) \in \mathcal{S}_V} V(y_p, y_q, \boldsymbol{x}_{pq}).$$

Here U is a unary potential function, and to make the exposition more general, we consider multiple types of unary potentials with $\mathcal{U}$ the set of all such unary potentials. $\mathcal{N}_U$ is a set of nodes for the potential U. Likewise, V is a pairwise potential function with $\mathcal{V}$ the set of all types of pairwise potential. $\mathcal{S}_V$ is the set of edges for the potential V. $\boldsymbol{x}_p$ and $\boldsymbol{x}_{pq}$ indicates the corresponding image regions which associate to the specified node and edge.

3.1. Unary potential functions

We formulate the unary potential function by stacking the FeatMap-Net for generating feature maps and a shallow fully connected network (referred to as Unary-Net) to generate the final output of the unary potential function. The unary potential function is written as follows:

$$U(y_p, \boldsymbol{x}_p; \boldsymbol{\theta}_U) = -z_{p,y_p}(\boldsymbol{x}; \boldsymbol{\theta}_U). \quad (2)$$

Here z_{p,y_p} is the output value of Unary-Net, which corresponds to the p-th node and the y_p-th class.

Fig. 3 includes an illustration of the Unary-Net and how

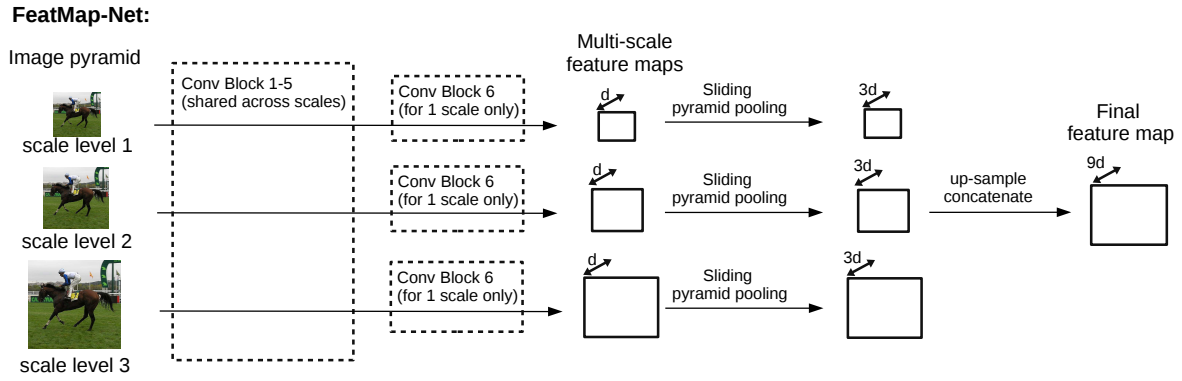


Figure 4. The details of our FeatMap-Net. An input image is first resized into 3 scales, then each resized image goes through 6 convolution blocks to output one feature map. Top 5 convolution blocks are shared for all scales. Every scale has a specific convolution block (Conv Block 6). We perform 2-level sliding pyramid pooling (see Fig. 5 for details). d indicates the feature dimension.

it integrates with FeatMap-Net. The unary potential at each CRF node is simply the K-dimensional output (where K is the number of classes) of Unary-Net applied to the node feature vector from the correpsonding location in the feature map (i.e. the output of FeatMap-Net).

3.2. Pairwise potential functions

Fig. 3 likewise illustrates how the pairwise potentials are generated. The edge features are formed by concatenating the corresponding feature vectors of two connected nodes (similar to [23]). The feature vector for each node in the pair is from the feature map output by FeatMap-Net. The edge features of one pair are then fed to a shallow fully connected network (referred to as Pairwise-Net) to generate the final output that is the pairwise potential. The size of this is $K \times K$ to match the number of possible label combinations for a pair of nodes. The pairwise potential function is written as follows:

$$V(y_p, y_q, \boldsymbol{x}_{pq}; \boldsymbol{\theta}_V) = -z_{p,q,y_p,y_q}(\boldsymbol{x}; \boldsymbol{\theta}_V). \quad (3)$$

Here z_{p,q,y_p,y_q} is the output value of Pairwise-Net. It is the confidence value for the node pair (p, q) when they are labeled with the class value (y_p, y_q), which measures the compatibility of the label pair (y_p, y_q) given the input image $\boldsymbol{x}$. $\boldsymbol{\theta}_V$ is the corresponding set of CNN parameters for the potential V, which we need to learn.

Our formulation of pairwise potentials is different from the Potts-model-based formulation in the existing methods of [3, 48]. The Potts-model-based pairwise potentials are a log-linear functions and employ a special formulation for enforcing neighborhood smoothness. In contrast, our pairwise potentials model the semantic compatibility between two nodes with the output for every possible value of the label pair (y_p, y_q) individually parameterized by CNNs.

In our system, after obtaining the coarse level prediction, we still need to perform a refinement step to obtain the final high-resolution prediction (as shown in Fig. 1). Hence we also apply the dense CRF method [24], as in many other recent methods, in the prediction refinement step. Therefore, our system takes advantage of both contextual CNN potentials and the traditional smoothness potentials to improve the final system. More details are described in Sec. 5.

As in [47, 20], modeling asymmetric relations requires the potential function is capable of modeling input orders, since we have: $V(y_p, y_q, \boldsymbol{x}_{pq}) \neq V(y_q, y_p, \boldsymbol{x}_{qp})$. Take the asymmetric relation "above/below" as an example; we take advantage of the input pair order to indicate the spatial configuration of two nodes, thus the input $(y_p, y_q, \boldsymbol{x}_{pq})$ indicates the configuration that the node p is spatially lies above the node q.

The asymmetric property is readily achieved with our general formulation of pairwise potentials. The potential output for every possible pairwise label combination for (p, q) is individually parameterized by the pairwise CNNs.

4. Exploiting background context

To encode rich background information, we use multi-scale CNNs and sliding pyramid pooling [26] for our FeatMap-Net. Fig. 4 shows the details of the FeatMap-Net.

CNNs with multi-scale image network inputs have shown good performance in some recent segmentation methods [13, 33]. The traditional pyramid pooling (in a sliding manner) on the feature map is able to capture information from background regions of different sizes. We observe that these two techniques (multi-scale network design and pyramid pooling) for encoding background information are very effective for improving performance.

Applying CNNs on multi-scale images has shown good performance in some recent segmentation methods [13, 33]. In our multi-scale network, an input image is first resized into 3 scales, then each resized image goes through 6 convolution blocks to output one feature map. In our experiment, the 3 scales for the input image are set to 1.2, 0.8 and 0.4. All scales share the same top 5 convolution blocks. In addition, each scale has an exclusive convolution block ("Conv

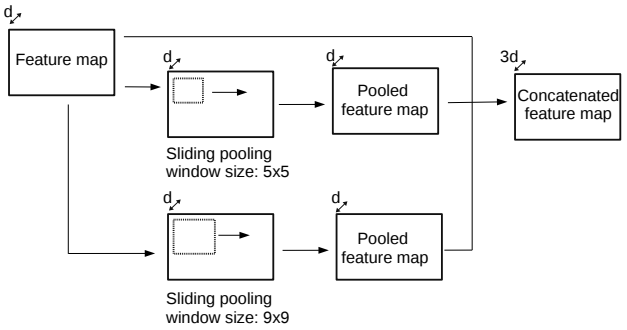


Figure 5. Details for sliding pyramid pooling. We perform 2-level sliding pyramid pooling on the feature map for capturing patch-background context, which encode rich background information and increase the field-of-view for the feature map.

Block 6" in the figure) which captures scale-dependent information. The resulting 3 feature maps (corresponding to 3 scales) are of different resolutions, therefore we upscale the two smaller ones to the size of the largest feature map using bilinear interpolation. These feature maps are then concatenated to form one feature map.

We perform spatial pyramid pooling [26] (a modified version using sliding windows) on the feature map to capture information from background regions in multiple sizes. This increases the field-of-view for the feature map and thus it is able to capture the information from a large image region. Increasing the field-of-view generally helps to improve performance [3].

The details of spatial pyramid pooling are illustrated in Fig. 5. In our experiment, we perform 2-level pooling for each image scale. We define 5×5 and 9×9 sliding pooling windows (max-pooling) to generate 2 sets of pooled feature maps, which are then concatenated to the original feature map to construct the final feature map.

The detailed network layer configuration for all networks are described in Fig. 6.

5. Prediction

In the prediction stage, our deep structured model will generate low-resolution prediction (as shown in Fig. 1), which is $1/16$ of the input image size. This is due to the stride setting of pooling or convolution layers for subsampling. Therefore, we apply two prediction stages for obtaining the final high-resolution prediction: the coarse-level prediction stage and the prediction refinement stage.

5.1. Coarse-level prediction stage

We perform CRF inference on our contextual structured model to obtain the coarse prediction of a test image. We consider the marginal inference over nodes for prediction:

$$\forall p \in \mathcal{N}: \quad P(y_p|\boldsymbol{x}) = \textstyle\sum_{\boldsymbol{y} \setminus y_p} P(\boldsymbol{y}|\boldsymbol{x}). \tag{4}$$

The obtained marginal distribution can be further applied in the next prediction stage for boundary refinement.

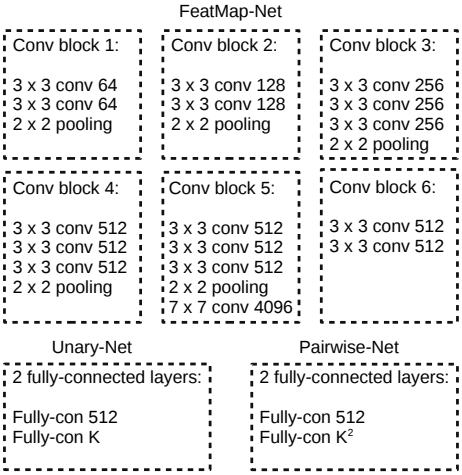


Figure 6. The detailed configuration of the networks: FeatMap-Net, Unary-Net and Pairwise-Net. K is the number of classes. For FeatMap-Net, the top 5 convolution blocks share the same configuration as the convolution blocks in the VGG-16 network. The stride of the last max pooling layer is 1, and for the other max pooling layers we use the same stride setting as VGG-16.

Our CRF graph does not form a tree structure, nor are the potentials submodular, hence we need to an apply approximate inference. To address this we apply an efficient message passing algorithm which is based on the mean field approximation [36]. The mean field algorithm constructs a simpler distribution $Q(\boldsymbol{y})$, e.g., a product of independent marginals: $Q(\boldsymbol{y}) = \prod_{p \in \mathcal{N}} Q_p(y_p)$, which minimizes the KL-divergence between the distribution $Q(\boldsymbol{y})$ and $P(\boldsymbol{y})$. In our experiments, we perform 3 mean field iterations.

5.2. Prediction refinement stage

We generate the score map for the coarse prediction from the marginal distribution which we obtain from the mean-field inference. We first bilinearly up-sample the score map of the coarse prediction to the size of the input image. Then we apply a common post-processing method [24] (dense CRF) to sharpen the object boundary for generating the final high-resolution prediction. This post-processing method leverages low-level pixel intensity information (color contrast) for boundary refinement. Note that most recent work on image segmentation similarly produces low-resolution prediction and have a upsampling and refinement process/model for the final prediction, e.g., [3, 48, 5].

In summary, we simply perform bilinear upsampling of the coarse score map and apply the boundary refinement post-processing. We argue that this stage can be further improved by applying more sophisticated refinement methods, e.g., training deconvolution networks [35], training multiple coarse to fine learning networks [9], and exploring middle layer features for high-resolution prediction [18, 32]. It is expected that applying better refinement approaches will gain further performance improvement.

6. CRF training

A common approach for CRF learning is to maximize the likelihood, or equivalently minimize the negative log-likelihood, which can be written for one image as:

$$-\log P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta}) = E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta}) + \log Z(\boldsymbol{x};\boldsymbol{\theta}). \quad (5)$$

Adding regularization to the CNN parameter $\boldsymbol{\theta}$, the optimization problem for CRF learning is:

$$\min_{\boldsymbol{\theta}} \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 + \sum_{i=1}^{N}\Big[E(\boldsymbol{y}^{(i)},\boldsymbol{x}^{(i)};\boldsymbol{\theta}) + \log Z(\boldsymbol{x}^{(i)};\boldsymbol{\theta})\Big]. \quad (6)$$

Here $\boldsymbol{x}^{(i)}$, $\boldsymbol{y}^{(i)}$ denote the i-th training image and its segmentation mask; N is the number of training images; λ is the weight decay parameter. We can apply stochastic gradient (SGD) based methods to optimize the above problem for learning $\boldsymbol{\theta}$. The energy function $E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta})$ is constructed from CNNs, and its gradient $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta})$ easily computed by applying the chain rule as in conventional CNNs. However, the partition function Z brings difficulties for optimization. Its gradient is:

$$\begin{aligned}
&\nabla_{\boldsymbol{\theta}} \log Z(\boldsymbol{x};\boldsymbol{\theta}) \\
&= \sum_{\boldsymbol{y}} \frac{\exp[-E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta})]}{\sum_{\boldsymbol{y}'}\exp[-E(\boldsymbol{y}',\boldsymbol{x};\boldsymbol{\theta})]} \nabla_{\boldsymbol{\theta}}[-E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta})] \\
&= -\mathbb{E}_{\boldsymbol{y}\sim P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})} \nabla_{\boldsymbol{\theta}} E(\boldsymbol{y},\boldsymbol{x};\boldsymbol{\theta}) \quad (7)
\end{aligned}$$

Generally the size of the output space $\mathcal{Y}$ is exponential in the number of nodes, which prohibits the direct calculation of Z and its gradient. The CRF graph we considered for segmentation here is a loopy graph (not tree-structured), for which the inference is generally computationally expensive. More importantly, usually a large number of SGD iterations (tens or hundreds of thousands) are required for training CNNs. Thus performing inference at each SGD iteration is very computationally expensive.

6.1. Piecewise training of CRFs

Instead of directly solving the optimization in (6), we propose to apply an approximate CRF learning method. In the literature, there are two popular types of learning methods which approximate the CRF objective : pseudo-likelihood learning [1] and piecewise learning [43]. The main advantage of these methods in term of training deep CRF is that they do not involve marginal inference for gradient calculation, which significantly improves the efficiency of training. Decision tree fields [37] and regression tree fields [22] are based on pseudo-likelihood learning, while piecewise learning has been applied in the work [43, 23].

Here we develop this idea for the case of training the CRF with the CNN potentials. In piecewise training, the conditional likelihood is formulated as a number of independent likelihoods defined on potentials, written as:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{U\in\mathcal{U}}\prod_{p\in\mathcal{N}_U} P_U(y_p|\boldsymbol{x}) \prod_{V\in\mathcal{V}}\prod_{(p,q)\in\mathcal{S}_V} P_V(y_p,y_q|\boldsymbol{x}).$$

The likelihood $P_U(y_p|\boldsymbol{x})$ is constructed from the unary potential U. Likewise, $P_V(y_p,y_q|\boldsymbol{x})$ is constructed from the pairwise potential V. P_U and P_V are written as:

$$P_U(y_p|\boldsymbol{x}) = \frac{\exp[-U(y_p,\boldsymbol{x}_p)]}{\sum_{y'_p}\exp[-U(y'_p,\boldsymbol{x}_p)]}, \quad (8)$$

$$P_V(y_p,y_q|\boldsymbol{x}) = \frac{\exp[-V(y_p,y_q,\boldsymbol{x}_{pq})]}{\sum_{y'_p,y'_q}\exp[-V(y'_p,y'_q,\boldsymbol{x}_{pq})]}. \quad (9)$$

Thus the optimization for piecewise training is to minimize the negative log likelihood with regularization:

$$\begin{aligned}
\min_{\boldsymbol{\theta}} \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 - \sum_{i=1}^{N}\Big[&\sum_{U\in\mathcal{U}}\sum_{p\in\mathcal{N}_U^{(i)}} \log P_U(y_p|\boldsymbol{x}^{(i)};\boldsymbol{\theta}_U) \\
&+ \sum_{V\in\mathcal{V}}\sum_{(p,q)\in\mathcal{S}_V^{(i)}} \log P_V(y_p,y_q|\boldsymbol{x}^{(i)};\boldsymbol{\theta}_V)\Big]. \quad (10)
\end{aligned}$$

Compared to the objective in (6) for direct maximum likelihood learning, the above objective does not involve the global partition function $Z(\boldsymbol{x};\boldsymbol{\theta})$. To calculate the gradient of the above objective, we only need to calculate the gradient $\nabla_{\boldsymbol{\theta}_U}\log P_U$ and $\nabla_{\boldsymbol{\theta}_V}\log P_V$. With the definition in (8), P_U is a conventional Softmax normalization function over only K (the number of classes) elements. Similar analysis can also be applied to P_V. Hence, we can easily calculate the gradient without involving expensive inference. Moreover, we are able to perform parallel training of potential functions, since the above objective is formulated as a summation of independent log-likelihoods.

As previously discussed, CNN training usually involves a large number of gradient update iterations. However this means that expensive inference during every gradient iteration becomes impractical. Our piecewise approach here provides a practical solution for learning CRFs with CNN potentials on large-scale data.

7. Experiments

We evaluate our method on 4 popular semantic segmentation datasets: PASCAL VOC 2012, NYUDv2, PASCAL-Context and SIFT-flow. The segmentation performance is measured by the intersection-over-union (IoU) score [12], the pixel accuracy and the mean accuracy [32].

The first 5 convolution blocks and the first convolution layer in the 6th convolution block are initialized from the VGG-16 network [42]. All remaining layers are randomly initialized. All layers are trained using back-propagation/SGD. As illustrated in Fig. 2, we use 2 types of pairwise potential functions. In total, we have 1 type of unary potential function and 2 types of pairwise potential functions. We formulate one specific FeatMap-Net and po-

Table 1. Segmentation results on NYUDv2 dataset (40 classes). We compare to a number of recent methods. Our method significantly outperforms the existing methods.

method	training data	pixel accuracy	mean accuracy	IoU
Gupta et al. [16]	RGB-D	60.3	-	28.6
FCN-32s [32]	RGB	60.0	42.2	29.2
FCN-HHA [32]	RGB-D	65.4	46.1	34.0
ours	RGB	**70.0**	**53.6**	**40.6**

tential network (Unary-Net or Pairwise-Net) for one type of potential function. We apply simple data augmentation in the training stage; specifically, we perform random scaling (from 0.7 to 1.2) and flipping of the images for training. Our system is built on MatConvNet [46].

7.1. Results on NYUDv2

We first evaluate our method on the dataset NYUDv2 [41]. NYUDv2 dataset has 1449 RGB-D images. We use the segmentation labels provided in [15] in which labels are processed into 40 classes. We use the standard training set which contains 795 images and the test set which contains 654 images. We train our models only on RGB images without using the depth information.

Results are shown in Table 1. Unless otherwise specified, our models are initialized using the VGG-16 network. VGG-16 is also used in the competing method FCN [32]. Our contextual model with CNN pairwise potentials achieves the best performance, which sets a new state-of-the-art result on the NYUDv2 dataset. Note that we do not use any depth information in our model.

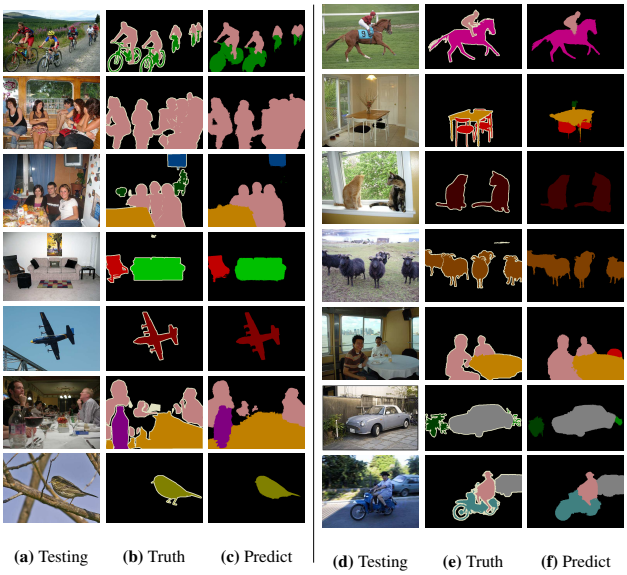
Component Evaluation We evaluate the performance contribution of different components of the FeatMap-Net for capturing patch-background context on the NYUDv2 dataset. We present the results of adding different components of FeatMap-Net in Table 2. We start from a baseline setting of our FeatMap-Net ("FullyConvNet Baseline" in the result table), for which multi-scale and sliding pooling is removed. This baseline setting is the conventional fully convolution network for segmentation, which can be considered as our implementation of the FCN method in [32]. The result shows that our CNN baseline implementation ("FullyConvNet") achieves very similar performance (slightly better) than the FCN method. Applying multi-scale network design and sliding pyramid pooling significantly improve the performance, which clearly shows the benefits of encoding rich background context in our approach. Applying the dense CRF method [24] for boundary refinement gains further improvement. Finally, adding our contextual CNN pairwise potentials brings significant further improvement, for which we achieve the best performance in this dataset.

7.2. Results on PASCAL VOC 2012

PASCAL VOC 2012 [12] is a well-known segmentation evaluation dataset which consists of 20 object categories

Table 2. Ablation Experiments. The table shows the value added by the different system components of our method on the NYUDv2 dataset (40 classes).

method	pixel accuracy	mean accuracy	IoU
FCN-32s [32]	60.0	42.2	29.2
FullyConvNet Baseline	61.5	43.2	30.5
+ sliding pyramid pooling	63.5	45.3	32.4
+ multi-scales	67.0	50.1	37.0
+ boundary refinement	68.5	50.9	38.3
+ CNN contextual pairwise	70.0	53.6	40.6



(a) Testing **(b)** Truth **(c)** Predict **(d)** Testing **(e)** Truth **(f)** Predict

Figure 7. Some prediction examples of our method.

and one background category. This dataset is split into a training set, a validation set and a test set, which respectively contain 1464, 1449 and 1456 images. Following a conventional setting in [19, 3], the training set is augmented by extra annotated VOC images provided in [17], which results in 10582 training images. We verify our performance on the PASCAL VOC 2012 test set. We compare with a number of recent methods with competitive performance. Since the ground truth labels are not available for the test set, we report the result through the VOC evaluation server.

The results of IoU scores are shown in the last column of Table 3. We first train our model only using the VOC images. We achieve 75.3 IoU score which is the best result amongst methods that only use the VOC training data.

To improve the performance, following the setting in recent work [3, 5], we train our model with the extra images from the COCO dataset [27]. With these extra training images, we achieve an IoU score of 77.2.

For further improvement, we also exploit the the middle-layer features as in the recent methods [3, 32, 18]. We learn extra refinement layers on the feature maps from middle layers to refine the coarse prediction. The feature maps from the middle layers encode lower level visual information which helps to predict details in the object boundaries. Specifically, we add 3 refinement convolution layers on top

Table 3. Individual category results on the PASCAL VOC 2012 test set (IoU scores). Our method performs the best

method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	potted	sheep	sofa	train	tv	mean
Only using VOC training data																					
FCN-8s [32]	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	62.2
Zoom-out [33]	85.6	37.3	**83.2**	62.5	66.0	85.1	80.7	84.9	27.2	73.2	57.5	78.1	79.2	81.1	77.1	53.6	74.0	49.2	71.7	63.3	69.6
DeepLab [3]	84.4	54.5	81.5	63.6	65.9	85.1	79.1	83.4	30.7	74.1	59.8	79.0	76.1	83.2	80.8	59.7	82.2	50.4	73.1	63.7	71.6
CRF-RNN [48]	87.5	39.0	79.7	64.2	68.3	87.6	80.8	84.4	30.4	78.2	60.4	80.5	77.8	83.1	80.6	59.5	82.8	47.8	78.3	67.1	72.0
DeconvNet [35]	89.9	39.3	79.7	63.9	68.2	87.4	81.2	86.1	28.5	77.0	62.0	79.0	80.3	83.6	80.2	58.8	**83.4**	54.3	80.7	65.0	72.5
DPN [31]	87.7	**59.4**	78.4	64.9	70.3	89.3	83.5	86.1	31.7	79.9	**62.6**	81.9	80.0	83.5	82.3	60.5	83.2	53.4	77.9	65.0	74.1
ours	**90.6**	37.6	80.0	**67.8**	**74.4**	**92.0**	**85.2**	**86.2**	**39.1**	**81.2**	58.9	**83.8**	**83.9**	**84.3**	**84.8**	**62.1**	83.2	**58.2**	**80.8**	**72.3**	**75.3**
Using VOC+COCO training data																					
DeepLab [3]	89.1	38.3	88.1	63.3	69.7	87.1	83.1	85.0	29.3	76.5	56.5	79.8	77.9	85.8	82.4	57.4	84.3	54.9	80.5	64.1	72.7
CRF-RNN [48]	90.4	55.3	88.7	68.4	69.8	88.3	82.4	85.1	32.6	78.5	64.4	79.6	81.9	**86.4**	81.8	58.6	82.4	53.5	77.4	70.1	74.7
BoxSup [5]	89.8	38.0	**89.2**	**68.9**	68.0	89.6	83.0	87.7	34.4	83.6	**67.1**	81.5	83.7	85.2	83.5	58.6	84.9	55.8	**81.2**	70.7	75.2
DPN [31]	89.0	**61.6**	87.7	66.8	74.7	91.2	**84.3**	87.6	36.5	86.3	66.1	84.4	87.8	85.6	85.4	63.6	87.3	61.3	79.4	66.4	77.5
ours+	**94.1**	40.7	84.1	67.8	**75.9**	**93.4**	**84.3**	**88.4**	**42.5**	**86.4**	64.7	**85.4**	**89.0**	85.8	**86.0**	**67.5**	**90.2**	**63.8**	80.9	**73.0**	**78.0**

Table 4. Segmentation results on PASCAL-Context dataset (60 classes). Our method performs the best.

method	pixel accuracy	mean accuracy	IoU
O2P [2]	-	-	18.1
CFM [6]	-	-	34.4
FCN-8s [32]	65.9	46.5	35.1
BoxSup [5]	-	-	40.5
ours	**71.5**	**53.9**	**43.3**

Table 5. Segmentation results on SIFT-flow dataset (33 classes). Our method performs the best.

method	pixel accuracy	mean accuracy	IoU
Liu et al. [28]	76.7	-	-
Tighe et al. [44]	75.6	41.1	-
Tighe et al. (MRF) [44]	78.6	39.2	-
Farabet et al. (balance) [13]	72.3	50.8	-
Farabet et al. [13]	78.5	29.6	-
Pinheiro et al. [38]	77.7	29.8	-
FCN-16s [32]	85.2	51.7	39.5
ours	**88.1**	**53.4**	**44.9**

of the feature maps from the first 5 max-pooling layers and the input image. The resulting feature maps and the coarse prediction score map are then concatenated and go through another 3 refinement convolution layers to output the refined prediction. The resolution of the prediction is increased from $1/16$ (coarse prediction) to $1/4$ of the input image. With this refined prediction, we further perform boundary refinement [24] to generate the final prediction. Finally, we achieve an IoU score of 78.0, *which is best reported result on this challenging dataset.* [1]

The results for each category are shown in Table 3. We outperform competing methods in most categories. For only using the VOC training set, our method outperforms the second best method, DPN [31], on 18 categories out of 20. Using VOC+COCO training set, our method outperforms DPN [31] on 15 categories out of 20. Some prediction examples of our method are shown in Fig. 7.

7.3. Results on PASCAL-Context

The PASCAL-Context [34] dataset provides the segmentation labels of the whole scene (including the "stuff" labels) for the PASCAL VOC images. We use the segmentation labels which contain 60 classes (59 classes plus the " background" class) for evaluation. We use the provided training/test splits. The training set contains 4998 images and the test set has 5105 images.

Results are shown in Table 4. Our method significantly outperforms the competing methods. To our knowledge, *ours is the best reported result on this dataset.*

7.4. Results on SIFT-flow

We further evaluate our method on the SIFT-flow dataset. This dataset contains 2688 images and provide the segmentation labels for 33 classes. We use the standard split for training and evaluation. The training set has 2488 images and the rest 200 images are for testing. Since images are in small sizes, we upscale the image by a factor of 2 for training. Results are shown in Table 5. We achieve the best performance for this dataset.

8. Conclusions

We have proposed a method which combines CNNs and CRFs to exploit complex contextual information for semantic image segmentation. We formulate CNN based pairwise potentials for modeling semantic relations between image regions. Our method shows best performance on several popular datasets including the PASCAL VOC 2012 dataset. The proposed method is potentially widely applicable to other vision tasks.

Acknowledgments This research was supported by the Data to Decisions Cooperative Research Centre and by the Australian Research Council through the Australian Centre for Robotic Vision (CE140100016). C. Shen's participation was supported by an ARC Future Fellowship (FT120100969). I. Reid's participation was supported by an ARC Laureate Fellowship (FL130100102).

C. Shen is the corresponding author (e-mail: chunhua.shen@adelaide.edu.au).

[1] The result link at the VOC evaluation server: http://host.robots.ox.ac.uk:8080/anonymous/XTTRFF.html

References

[1] J. Besag. Efficiency of pseudolikelihood estimation for simple Gaussian fields. *Biometrika*, 1977. 6

[2] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *Proc. Eur. Conf. Comp. Vis.*, 2012. 8

[3] L. 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. Learning Representations*, 2015. 1, 2, 4, 5, 7, 8

[4] L.-C. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun. Learning deep structured models. In *Proc. Int. Conf. Machine Learn.*, 2015. 3

[5] J. Dai, K. He, and J. Sun. BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proc. Int. Conf. Comp. Vis.*, 2015. 2, 5, 7, 8

[6] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2015. 8

[7] C. Doersch, A. Gupta, and A. A. Efros. Context as supervisory signal: Discovering objects with predictable context. In *Proc. European Conf. Computer Vision*, 2014. 2

[8] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *Proc. Eur. Conf. Comp. Vis.*, 2014. 2

[9] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015. 2, 5

[10] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *Proc. Int. Conf. Comp. Vis.*, 2013. 2

[11] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Proc. Adv. Neural Info. Process. Syst.*, 2014. 2

[12] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. In *Proc. Int. J. Comp. Vis.*, 2010. 6, 7

[13] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE T. Pattern Analysis & Machine Intelligence*, 2013. 1, 2, 4, 8

[14] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2014. 2

[15] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2013. 7

[16] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *Proc. Eur. Conf. Comp. Vis.*, 2014. 7

[17] B. Hariharan, P. Arbelaez, L. D. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *Proc. Int. Conf. Comp. Vis.*, 2011. 7

[18] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2014. 2, 5, 7

[19] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *Proc. European Conf. Computer Vision*, 2014. 2, 7

[20] D. Heesch and M. Petrou. Markov random fields with asymmetric interactions for modelling spatial context in structured scene labelling. *Journal of Signal Processing Systems*, 2010. 4

[21] G. Heitz and D. Koller. Learning spatial context: Using stuff to find things. In *Proc. European Conf. Computer Vision*, 2008. 1, 2

[22] J. Jancsary, S. Nowozin, T. Sharp, and C. Rother. Regression tree fieldsan efficient, non-parametric approach to image labeling problems. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2012. 6

[23] A. Kolesnikov, M. Guillaumin, V. Ferrari, and C. H. Lampert. Closed-form training of conditional random fields for large scale image segmentation. In *Proc. Eur. Conf. Comp. Vis.*, 2014. 4, 6

[24] P. Krähenbühl and V. Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Proc. Adv. Neural Info. Process. Syst.*, 2012. 2, 4, 5, 7, 8

[25] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *Proc. Int. Conf. Mach. Learn.*, 2013. 2

[26] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2006. 2, 4, 5

[27] 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 *Proc. Eur. Conf. Comp. Vis.*, 2014. 7

[28] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE T. Pattern Analysis & Machine Intelligence*, 2011. 8

[29] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2015. 2, 3

[30] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields, 2015. http://arxiv.org/abs/1502.07411. 2, 3

[31] Z. Liu, X. Li, P. Luo, C. C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *Proc. Int. Conf. Comp. Vis.*, 2015. 8

[32] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2015. 1, 2, 5, 6, 7, 8

[33] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2015. 2, 4, 8

[34] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, et al. The role of context for object detection and semantic segmentation in the wild. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2014. 8

[35] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proc. Int. Conf. Comp. Vis.*, 2015. 2, 5, 8

[36] S. Nowozin and C. Lampert. Structured learning and prediction in computer vision. *Found. Trends. Comput. Graph. Vis.*, 2011. 5

[37] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *Proc. Int. Conf. Comp. Vis.*, 2011. 6

[38] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene parsing. In *Proc. Int. Conf. Machine Learn.*, 2014. 8

[39] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *Proc. Int. Conf. Comp. Vis.*, 2007. 2

[40] A. G. Schwing and R. Urtasun. Fully connected deep structured networks, 2015. http://arxiv.org/abs/1503.02351. 2

[41] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Proc. Eur. Conf. Comp. Vis.*, 2012. 7

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. Int. Conf. Learning Representations*, 2015. 6

[43] C. A. Sutton and A. McCallum. Piecewise training for undirected models. In *Proc. Conf. Uncertainty Artificial Intelli*, 2005. 2, 6

[44] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2013. 8

[45] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Proc. Adv. Neural Info. Process. Syst.*, 2014. 3

[46] A. Vedaldi and K. Lenc. MatConvNet – convolutional neural networks for matlab, 2014. 7

[47] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *Proc. IEEE Conf. Comp. Vis. Pattern Recogn.*, 2006. 4

[48] 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. In *Proc. Int. Conf. Comp. Vis.*, 2015. 2, 4, 5, 8