Gaussian Conditional Random Field Network for Semantic Segmentation

Raviteja Vemulapalli[†], Oncel Tuzel[*], Ming-Yu Liu[*], and Rama Chellappa[†]
[†]Center for Automation Research, UMIACS, University of Maryland, College Park.
[*]Mitsubishi Electric Research Laboratories, Cambridge, MA.

Abstract

In contrast to the existing approaches that use discrete Conditional Random Field (CRF) models, we propose to use a Gaussian CRF model for the task of semantic segmentation. We propose a novel deep network, which we refer to as Gaussian Mean Field (GMF) network, whose layers perform mean field inference over a Gaussian CRF. The proposed GMF network has the desired property that each of its layers produces an output that is closer to the maximum a posteriori solution of the Gaussian CRF compared to its input. By combining the proposed GMF network with deep Convolutional Neural Networks (CNNs), we propose a new end-to-end trainable Gaussian conditional random field network. The proposed Gaussian CRF network is composed of three sub-networks: (i) a CNN-based unary network for generating unary potentials, (ii) a CNN-based pairwise network for generating pairwise potentials, and (iii) a GMF network for performing Gaussian CRF inference. When trained end-to-end in a discriminative fashion, and evaluated on the challenging PASCALVOC 2012 segmentation dataset, the proposed Gaussian CRF network outperforms various recent semantic segmentation approaches that combine CNNs with discrete CRF models.

1. Introduction

Semantic segmentation, which aims to predict a category label for every pixel in the image, is an important task for scene understanding. Though it has received significant attention from the vision community over the past few years, it still remains a challenging problem due to large variations in the visual appearance of the semantic classes and complex interactions between various classes in the visual world. Recently, convolutional neural networks have been shown to work very well for this challenging task [35, 13, 18, 30, 31, 35]. Their success can be attributed to several factors such as their ability to represent complex input-output relationships, feed-forward nature of their inference, availability of large training datasets and fast computing hardware like GPUs, etc.

However, Convolutional Neural Networks (CNNs) may not be optimal for structured prediction tasks such as semantic segmentation as they do not model the interactions between output variables directly. Acknowledging this, various semantic segmentation approaches have been proposed in the recent past that use Conditional Random Field (CRF) models [26] on top of CNNs [3, 7, 13, 33, 37, 45, 55], and all these approaches have shown significant improvement in the segmentation results by using CRFs. By combining CNNs and CRFs, these approaches get the best of both worlds: the ability of CNNs to model complex input-output relationships and the ability of CRFs to directly model the interactions between output variables. While some of these approaches use CRF as a separate post-processing step [3, 7, 13, 33, 37], some other approaches train the CNNs along with the CRFs in an end-to-end fashion [45, 55].

All of the above approaches use discrete graphical models, and hence end up using graph-cuts or mean field-based approximate inference procedures. Though these inference procedures do not have global optimum guarantees, they have been successfully used for the semantic segmentation task in conjunction with CNNs. In contrast to discrete graphical models, Gaussian graphical models [41, 50] are simpler models, and have inference procedures that are guaranteed to converge to the global optimal solution. Gaussian graphical models have been used in the past for various applications such as image denoising [22, 50], depth estimation [29, 42], deblurring [43, 56], edge detection [54], texture classification [5], etc.

While a discrete CRF is a natural fit for labeling tasks such as semantic segmentation, one needs to use inference techniques that do not have optimality guarantees. While exact inference is tractable in the case of a Gaussian CRF, it is not clear if this model is a good fit for discrete labeling tasks. This leads us to the following question: *Should we use a better model with approximate inference or an approximate model with better inference?*

To answer this question, in this work, we use a Gaussian CRF (GCRF) model for the task of semantic segmentation. To use a GCRF model for this discrete labeling task, we first replace each discrete variable with a vector of K mutually

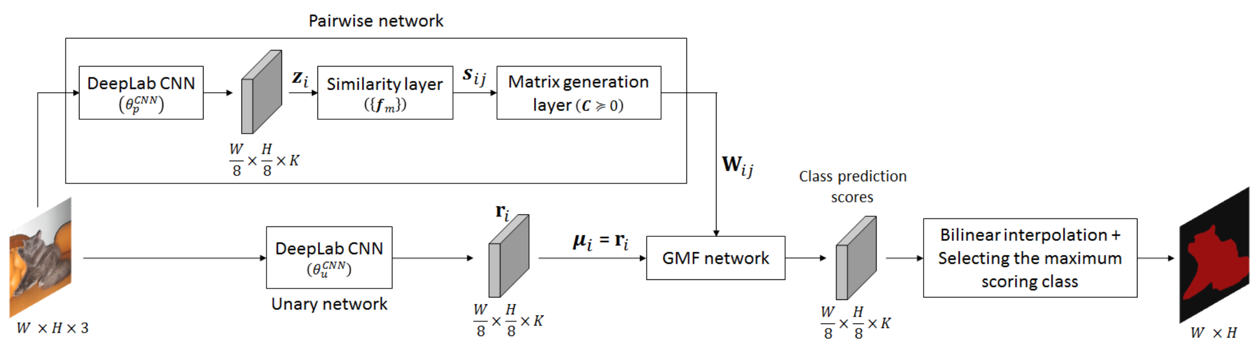
Pairwise network
DeepLab CNN (θ_p^{CNN})
z_i
Similarity layer ($\{f_m\}$)
s_{ij}
Matrix generation layer ($C \succcurlyeq 0$)
$\frac{W}{8} \times \frac{H}{8} \times K$
$\mathbf{W}_{ij}$
Class prediction scores
$\mathbf{r}_i$
$\mu_i = \mathbf{r}_i$
GMF network
Bilinear interpolation + Selecting the maximum scoring class
DeepLab CNN (θ_u^{CNN})
Unary network
$W \times H \times 3$
$\frac{W}{8} \times \frac{H}{8} \times K$
$\frac{W}{8} \times \frac{H}{8} \times K$
$W \times H$

Figure 1: Proposed GCRF network: The GMF network performs GCRF inference using the outputs of unary and pairwise networks. The output of GMF network is upsampled to full image resolution using bilinear interpolation. Note that the parameters of this GCRF network are the unary network parameters θ_u^{CNN} and the pairwise network parameters $\{\theta_p^{CNN}, \{\mathbf{f}_m\}, \mathbf{C} \succeq 0\}$.

exclusive binary variables, where K is the number of possible values the discrete variable can take, and then model all the variables jointly as a multivariate Gaussian by relaxing the mutual exclusivity and binary constraints. After the GCRF inference, the discrete label assignment is done based on which of the K corresponding variables has the maximum value.

Though the Maximum a Posteriori (MAP) solution can be obtained in closed form in the case of GCRFs, it involves solving a linear system with number of variables equal to the number of nodes in the graph times the dimensionality of node variables (which is equal to the number of spatial locations times the number of classes in the case of semantic segmentation). Solving such a large linear system could be computationally prohibitive, especially for dense graphs where each node is connected to several other nodes. Hence, in this work, instead of exactly solving a large linear system, we unroll a fixed number of Gaussian Mean Field (GMF) inference steps as layers of a deep network, which we refer to as GMF network. Note that the GMF inference is different from the mean field inference used in [24] for discrete CRFs with Gaussian edge potentials.

While GMF updates are guaranteed to give the MAP solution upon convergence, parallel updates are guaranteed to converge only under certain constraints such as diagonal dominance of the precision matrix of the joint Gaussian [53]. If the nodes are updated serially, then the GMF inference is equivalent to an alternating minimization approach in which each subproblem is solved optimally, and hence it will converge (as finding the MAP solution for a GCRF is a convex problem with a smooth cost function). But, using serial updates would be very slow when the number of variables is large. To avoid both these issues, in this work, we use a bipartite graph structure that allows us to update half of the nodes in parallel in each step without loosing the convergence guarantee even when the diagonal dominance constraint is not satisfied. Using this bipartite structure, we ensure that each layer of our GMF network produces an output that is closer to the MAP solution compared to its input.

By combining the proposed GMF network with CNNs, we propose a new end-to-end trainable deep network, which we refer to as Gaussian CRF network, for the task of semantic segmentation. The proposed GCRF network consists of a CNN-based unary network for generating the unary potentials, a CNN-based pairwise network for generating the pairwise potentials and a GMF network for performing the GCRF inference. Figure 1 gives an overview of the entire network. When trained discriminatively using the ImageNet and PASCALVOC data (ImageNet used for pretraining the CNNs), the proposed GCRF network gave a mean intersection-over-union (IOU) score of 73.2 on the challenging PASCALVOC 2012 test set [12], outperforming various recent approaches that combined CNNs with discrete CRFs. Also, when compared to just using the unary network, we improve the mean IOU by 6.2 points.

Contributions:

- **Gaussian CRF for semantic segmentation:** In contrast to the existing approaches that use discrete CRF models, we propose to use a GCRF model for the task of semantic segmentation. Compared to discrete CRFs, GCRFs are simpler models that can be solved optimally.
- **GMF network:** We propose a novel deep network by unfolding a fixed number of Gaussian mean field iterations. Using a bipartite graph structure, we ensure that each layer in our GMF network produces an output that is closer to the optimal solution compared to its input.
- **Gaussian CRF network:** We propose a new end-to-end trainable deep network that combines the GCRF model with CNNs for the task of semantic segmentation.
- **Results:** We show that the proposed GCRF network outperforms various existing discrete CRF-based approaches on the challenging PASCALVOC 2012 test set (when trained with ImageNet and PASCALVOC data).

2. Related Work

Semantic segmentation using CNNs: In the recent past, numerous semantic segmentation approaches have been proposed based on CNNs. In [14, 17], each region proposal was classified into one of the semantic classes by using CNN features. Instead of applying CNN to each region independently as in [14, 17], [9] applied the convolutional layers only once to the entire image, and generated region features by using pooling after the final convolutional layer.

Different from the above approaches, [13] trained a CNN to directly extract features at each pixel. To capture the information present at multiple scales, CNN was applied to the input image multiple times at different resolutions, and the features from all the resolutions were concatenated to get the final pixel features. This multiscale feature was then classified using a two-layer neural network. Finally, post-processing steps like CRF and segmentation tree were used to further improve the results. Building on top of these CNN features, [46, 47] introduced a recursive context propagation network that enriched the CNN features by adding image level contextual information. Instead of using a CNN multiple times, [7, 18, 31] proposed to use the features extracted by the intermediate layers of a deep CNN to capture the multi-scale information. Recently, [32] trained a deconvolution network for the task of semantic segmentation. This network was applied separately to each region proposal, and all the results were aggregated to get the final predictions.

Most of the CNN-based methods mentioned above use superpixels or region proposals, and hence the errors in the initial proposals will remain no matter how good the CNN features are. Different from these methods, [30] directly produced dense segmentation maps by upsampling the predictions produced by a CNN using a trainable deconvolution layer. To obtain finer details in the upsampled output, they combined the final layer predictions with predictions from lower layers.

Combining CNNs and CRFs for semantic segmentation: Though CNNs have been shown to work very well for the task of semantic segmentation, they may not be optimal as they do not model the interactions between the output variables directly, which is important for semantic segmentation. To overcome this issue, various recent approaches [3, 7, 13, 33, 37] have used discrete CRF [26] models on top of CNNs. While [13] defined a CRF on superpixels and used graph-cuts based inference, [3, 7, 33, 37] defined a CRF directly on image pixels and used the efficient mean field inference proposed in [24]. Instead of using CRF as a post-processing step, [55] trained a CNN along with a CRF in an end-to-end fashion by converting the mean field inference procedure of [24] into a recurrent neural network. Similar joint training strategy was also used in [45].

In all these approaches, the CRF edge potentials were designed using hand-chosen features like image gradients, pixel color values, spatial locations, etc. and the potential function parameters were manually tuned. Contrary to this, recently, [28] has learned both unary and pairwise potentials using CNNs. While all these approaches learn CNN-based potentials and use message passing algorithms to perform CRF inference, [27] has recently proposed to use CNNs to directly learn the messages in message passing inference.

The idea of jointly training a CNN and graphical model has also been used for other applications such as sequence labeling [10, 34], text recognition [21], human pose estimation [52], predicting words from images [6], handwritten word recognition [4]. Recently, various CNN-based semantic segmentation approaches have also been proposed for the semi and weakly supervised settings [8, 20, 33, 36].

Unrolling inference as a deep network: The proposed approach is also related to a class of algorithms that learn model parameters discriminatively by back-propagating the gradient through a fixed number of inference steps. In [2], the fields of experts [40] model was discriminatively trained for image denoising by unrolling a fixed number of gradient descent inference steps. In [6, 11, 25, 49] discrete graphical models were trained by back-propagating through either the mean field or the belief propagation inference iterations. In [39], message passing inference machines were trained by considering the belief propagation-based inference of a discrete graphical model as a sequence of predictors. In [15], a feed-forward sparse code predictor was trained by unrolling a coordinate descent-based sparse coding inference algorithm. In [19], a new kind of non-negative deep network was introduced by deep unfolding of non-negative factorization model. Different from these approaches, in this work, we unroll the mean filed inference of a GCRF model as a deep network, and train our CNN-based potential functions along with the GCRF inference network in an end-to-end fashion.

Gaussian conditional random fields: GCRFs [50] are popular models for structured inference tasks like denoising [22, 43, 44, 50, 56], deblurring [43, 44, 56], depth estimation [29, 42], etc., as they model continuous quantities and can be efficiently solved using linear algebra routines.

Gaussian CRF was also used for discrete labeling tasks earlier in [51], where a Logistic Random Field (LRF) was proposed by combining a quadratic model with logistic function. While the LRF used a logistic function on top of a GCRF to model the output, we directly model the output using a GCRF. Unlike [51], which used hand-chosen features like image gradients, color values, etc. to model the potentials, we use CNN-based potential functions.

Recently, [29] trained a CNN along with a GCRF model for image-based depth prediction. The GCRF model of [29] was defined on superpixels and had edges only between adjacent superpixels. As the resulting graph was sparse with few nodes, [29] performed exact GCRF inference by solving a linear system. In contrast, we define our GCRF model directly on top of the dense CNN output and connect each node to several neighbors. Since the number of variables in our GCRF model is very large, exactly solving a linear system would be computationally expensive. Hence, we unfold a fixed number of GMF inference steps into a deep network. Also, while [29] used hand-designed features like color histogram, local binary patterns, etc. for designing their pairwise potentials, we use CNN-based pairwise potentials.

The idea of combining the GCRF model with neural networks (one or two layers) has also been explored previously for applications such as document retrieval [38] and facial landmark detection [1]. However, the way we model our potential functions and perform inference is different from these works.

Notations: We use bold face small letters to denote vectors and bold face capital letters to denote matrices. We use $\mathbf{A}^\top$ and $\mathbf{A}^{-1}$ to denote the transpose and inverse of a matrix $\mathbf{A}$. We use $\|\mathbf{b}\|_2^2$ to denote the squared ℓ_2 norm of a vector $\mathbf{b}$. $\mathbf{A} \succeq 0$ means $\mathbf{A}$ is symmetric and positive semidefinite.

3. Gaussian Conditional Random Field Model

In semantic segmentation, we are interested in assigning each pixel in an image $\mathbf{X}$ to one of the K possible classes. As mentioned earlier, we use K variables (one for each class) to model the output at each pixel, and the final label assignment is done based on which of these K variables has the maximum value. Let $\mathbf{y}_i = [y_{i1}, \ldots, y_{iK}]$ be the vector of K output variables associated with the i^{th} pixel, and $\mathbf{y}$ be the vector of all output variables. In this work, we model the conditional probability density $P(\mathbf{y}|\mathbf{X})$ as a Gaussian distribution given by

$$\begin{aligned} &P(\mathbf{y}|\mathbf{X}) \propto \exp\left\{-\frac{1}{2}\,E(\mathbf{y}|\mathbf{X})\right\}, \text{ where} \\ &E\left(\mathbf{y}|\mathbf{X}\right) = \sum_i \|\mathbf{y}_i - \mathbf{r}_i(\mathbf{X};\theta_u)\|_2^2 \\ &\qquad + \sum_{ij} \left(\mathbf{y}_i - \mathbf{y}_j\right)^\top \mathbf{W}_{ij}\left(\mathbf{X};\theta_p\right)\left(\mathbf{y}_i - \mathbf{y}_j\right). \end{aligned} \tag{1}$$

The first term in the above energy function E is the unary term and the second term is the pairwise term [1]. Here, both $\mathbf{r}_i$ and $\mathbf{W}_{ij} \succeq 0$ are functions of the input image $\mathbf{X}$ with θ_u and θ_p being the respective function parameters. Note that when $\mathbf{W}_{ij} \succeq 0$ for all pairs of pixels, the unary and pairwise terms can be combined together into a single positive semidefinite quadratic form.

[1]Note that we have only one term for each pair of pixels, i.e., we do not have separate $\mathbf{W}_{ij}$ and $\mathbf{W}_{ji}$ for the pair (i, j). We just have one $\mathbf{W}$ for (i, j) which can be interpreted as both $\mathbf{W}_{ij}$ and $\mathbf{W}_{ji}$ based on the context.

The optimal $\mathbf{y}$ that minimizes the energy function E can be obtained in closed form since the minimization of E is an unconstrained quadratic program. However, this closed form solution involves solving a linear system with number of variables equal to the number of pixels times the number of classes. Since solving such a large linear system could be computationally prohibitive, in this work, we use the iterative mean field inference approach.

3.1. Gaussian mean field inference

The standard mean field approach approximates the joint distribution $P(\mathbf{y}|\mathbf{X})$ using a simpler distribution $Q(\mathbf{y}|\mathbf{X})$ which can be written as a product of independent marginals, i.e, $Q(\mathbf{y}|\mathbf{X}) = \prod_i Q_i(\mathbf{y}_i|\mathbf{X})$. [2] This approximate distribution is obtained by minimizing the KL-divergence between the distributions P and Q. In the case of Gaussian, the mean field approximation Q and the original distribution P have the same mean [53]. Hence, finding the MAP solution $\mathbf{y}$ is equivalent to finding the mean $\boldsymbol{\mu}$ of the distribution Q.

For the Gaussian distribution in (1), the mean field updates for computing the mean $\boldsymbol{\mu}$ are given by

$$\boldsymbol{\mu}_i \leftarrow \Big(I + \sum_j \mathbf{W}_{ij}\Big)^{-1}\Big(\mathbf{r}_i + \sum_j \mathbf{W}_{ij}\boldsymbol{\mu}_j\Big). \tag{2}$$

Here, $\boldsymbol{\mu}_i$ is the mean of marginal Q_i. Please refer to the supplementary material for detailed derivations. It is easy to see that if we use the standard alternating minimization approach (in which we update one pixel at a time) to find the optimal $\mathbf{y}$ that minimizes the energy function in (1), we would end up with the same update equation. Since the energy function is a convex quadratic in the case of GCRF and update (2) solves each subproblem optimally, i.e., finds the optimal $\mathbf{y}_i$ (or $\boldsymbol{\mu}_i$) when all the other $\mathbf{y}_j$ (or $\boldsymbol{\mu}_j$) are fixed, performing serial updates is guaranteed to give us the MAP solution. However, it would be very slow since we are dealing with a large number of variables.

While using parallel updates seems to be a reasonable alternative, convergence of parallel updates is guaranteed only under certain constraints like diagonal dominance of the precision matrix of the distribution P [53]. Imposing such constraints could restrict the model capacity in practice. For example, in our GCRF model (1), we can satisfy the diagonal dominance constraint by making all $\mathbf{W}_{ij}$ diagonal. However, this can be very restrictive, as making the non-diagonal entries of $\mathbf{W}_{ij}$ zero will remove the direct inter-class interactions between pixels i and j, i.e., there will not be any interaction term in the energy function between the variables y_{ip} and y_{jq} for $p \neq q$.

[2]Note that instead of using marginals of scalar variables y_{ik}, we are using marginals of vector variables $\mathbf{y}_i$.

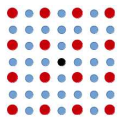

Figure 2: Each pixel in our CRF is connected to every other pixel along both rows and columns within a spatial neighborhood. Here, all the pixels that are connected to the center black pixel are shown in red. If the black pixel is on odd column, all the pixels connected to it will be on even columns and vice versa.

3.2. Bipartite graph structure for parallel updates

While we want to avoid the diagonal dominance constraint, we also want to update as many variables as possible in parallel. To address this problem, we use a bipartite graph structure, which allows us to update half of the variables in parallel in each step, and still guarantees convergence without the diagonal dominance constraint.

Note that our graphical model has a node for each pixel, and each node represents a vector of K variables. In order to update the i^{th} node using (2), we need to keep all the other nodes connected to the i^{th} node (i.e., all the nodes with non-zero $\mathbf{W}_{ij}$) fixed. If we partition the image into odd and even columns (or odd and even rows) and avoid edges within the partitions, then we can optimally update all the odd columns (or rows) in parallel using (2) while keeping the even columns (or rows) fixed and vice versa. This is again nothing but an alternating minimization approach in which each subproblem (corresponding to half of the nodes in the graph) is optimally solved, and hence is guaranteed to converge to the global optimum (since we are dealing with a convex problem).

Generally when using graphical models, each pixel is connected to all the pixels within a spatial neighborhood. In this work, instead of using all the neighbors, we connect each pixel to every other neighbor along both rows and columns. Figure 2 illustrates this for a 7×7 spatial neighborhood. It is easy to see that with this connectivity, we can partition the image into even and odd columns (or even and odd rows) without any edges within the partitions.

4. Gaussian CRF network

The proposed GCRF network consists of three components: *Unary network*, *Pairwise network* and *GMF network*. While the unary and pairwise networks generate the $\mathbf{r}_i$ and $\mathbf{W}_{ij}$ that are respectively used in the unary and pairwise terms of the energy function (1), the GMF network performs Gaussian mean field inference using the outputs of unary and pairwise networks. Figure 1 gives an overview of the proposed GCRF network.

Unary network: To generate the $\mathbf{r}_i$ used in the unary term of the energy function (1), we use the DeepLab-MSc-LargeFov network of [7] (along with the softmax layer), which is a modified version of the popular VGG-16 network [48]. Modifications compared to VGG-16 include converting the fully-connected layers into convolutional layers, skipping downsampling after the last two pooling layers, modifying the convolutional layers after the fourth pooling layer, and using multi-scale features similar to [18]. Please refer to [7] for further details. For brevity, we will refer to this DeepLab-MSc-LargeFov network as DeepLab CNN in the rest of the paper. We will denote the parameters of this unary DeepLab network using θ_u^{CNN}.

Pairwise network: Our pairwise network generates the matrices $\mathbf{W}_{ij}$ that are used in the pairwise term of the energy function (1). In this work, we compute each $\mathbf{W}_{ij}$ as

$$\mathbf{W}_{ij} = s_{ij}\mathbf{C}, \ \mathbf{C} \succeq 0, \tag{3}$$

where $s_{ij} \in [0, 1]$ is a measure of similarity between pixels i and j, and the learned matrix $\mathbf{C}$ encodes the class compatibility information. We compute the similarity measure s_{ij} using

$$s_{ij} = e^{-(\mathbf{z}_i - \mathbf{z}_j)^\top \mathbf{F} (\mathbf{z}_i - \mathbf{z}_j)}, \tag{4}$$

where $\mathbf{z}_i$ is the feature vector extracted at i^{th} pixel using a DeepLab CNN (with parameters θ_p^{CNN}), and the learned matrix $\mathbf{F} \succeq 0$ defines a Mahalanobis distance function. Note that the exponent of s_{ij} can be written as

$$(\mathbf{z}_i - \mathbf{z}_j)^\top \mathbf{F} (\mathbf{z}_i - \mathbf{z}_j) = \sum_{m=1}^{M} (\mathbf{f}_m^\top \mathbf{z}_i - \mathbf{f}_m^\top \mathbf{z}_j)^2, \tag{5}$$

where $\mathbf{F} = \sum_{m=1}^{M} \mathbf{f}_m \mathbf{f}_m^\top$. Hence, we implement the Mahalanobis distance computation as convolutions (of $\mathbf{z}_i$ with filters $\mathbf{f}_m$) followed by an Euclidean distance computation.

The overall pairwise network consists of a DeepLab CNN that generates the pixel features $\mathbf{z}_i$, a similarity layer that computes s_{ij} for every pair of connected pixels using (4) and (5), and a matrix generation layer that computes the matrices $\mathbf{W}_{ij}$ using (3). Note that here $\{\mathbf{f}_m\}$ are the parameters of the similarity layer and $\mathbf{C} \succeq 0$ are the parameters of the matrix generation layer.

GMF network: The proposed GMF network performs a fixed number of Gaussian mean field updates using the outputs of unary and pairwise networks. The input to the network is initialized using the unary output, $\mu^1 = \mathbf{r} = \{\mathbf{r}_i\}$. The network consists of several sequential GMF layers, where each GMF layer has two sub-layers (an even update layer followed by an odd update layer, See Figure 3):

- **Even update layer:** This sublayer takes the output of previous layer as input, and updates the even column nodes using (2) while keeping odd column nodes fixed.
- **Odd update layer:** This sublayer takes the output of even update layer as input, and updates the odd column nodes using (2) while keeping even column nodes fixed.

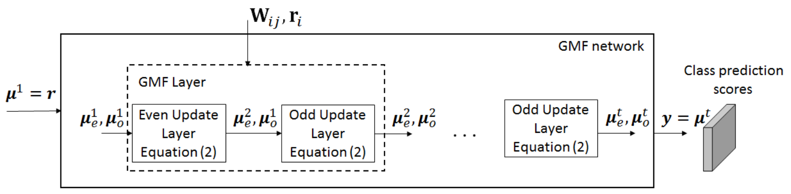
$\mathbf{W}_{ij}, \mathbf{r}_i$
GMF network
$\boldsymbol{\mu}^1 = \mathbf{r}$
GMF Layer
$\boldsymbol{\mu}_e^1, \boldsymbol{\mu}_o^1$
Even Update Layer Equation (2)
$\boldsymbol{\mu}_e^2, \boldsymbol{\mu}_o^1$
Odd Update Layer Equation (2)
$\boldsymbol{\mu}_e^2, \boldsymbol{\mu}_o^2$
Odd Update Layer Equation (2)
$\boldsymbol{\mu}_e^t, \boldsymbol{\mu}_o^t$
$\mathbf{y} = \boldsymbol{\mu}^t$
Class prediction scores

Figure 3: GMF Network. μ_e^t and μ_o^t are even and odd column nodes respectively where t indexes the layers, $\mu^t = \{\mu_e^t, \mu_o^t\}$. Network is initialized with unary network output $\mu^1 = \mathbf{r}$.

As explained in the previous section, because of the bipartite graph structure, the update performed by each of the above sublayers is an optimal update. Hence, each layer of our GMF network is guaranteed to generate an output that is closer to the MAP solution compared to its input (unless the input itself is the MAP solution, in which case the output will be equal to the input).

Combining the unary, pairwise and GMF networks, we get the proposed GCRF network, which can be trained in an end-to-end fashion. The parameters of the network are the unary network parameters $\theta_u = \theta_u^{CNN}$, and the pairwise network parameters $\theta_p = \{\theta_p^{CNN}, \{\mathbf{f}_m\}, \mathbf{C} \succeq 0\}$. Note that since we use a fixed number of layers in our GMF network, the final output is not guaranteed to be the MAP solution of our GCRF model. However, since we train the entire network discriminatively in an end-to-end fashion, the unary and pairwise networks would learn to generate appropriate $\mathbf{r}_i$ and $\mathbf{W}_{ij}$ such that the output after a fixed number of mean field updates would be close to the desired output.

Note that the DeepLab network has three downsampling layers, and hence the size of its output is 1/8 times the input image size. We apply our GCRF model to this low resolution output and upsample the GMF network output to the input image resolution by using bilinear interpolation.

Discrete label assignment: Note that the final output at each pixel is a K-dimensional vector where K is the number of classes. Let $\mathbf{y}_i^* = [y_{i1}^*, \dots, y_{iK}^*]$ be the final output at i^{th} pixel. Then the predicted class label of i^{th} pixel is given by argmax$_k$ y_{ik}^*.

Training loss function: For training the network, we use the following loss function at each pixel

$$L(\mathbf{y}_i^*, l_i) = -\min\left(0,\ y_{il_i}^* - \max_{k \neq l_i} y_{ik}^* - T\right), \quad (6)$$

where l_i is the true class label. This loss function basically encourages the output associated with the true class to be greater than the output associated with all the other classes by a margin T.

Training: We train the proposed GCRF network discriminatively by minimizing the above loss function. We use standard backpropagation to compute the gradient of the network parameters. Due to space constraints, we present the derivative formulas in the supplementary material. Note that we have a constrained optimization problem here due to the symmetry and positive semidefiniteness constraints on the parameter $\mathbf{C}$. We convert this constrained problem into an unconstrained one by parametrizing $\mathbf{C}$ as $\mathbf{C} = \mathbf{R}\mathbf{R}^\top$, where $\mathbf{R}$ is a lower triangular matrix, and use stochastic gradient descent for optimization.

5. Experiments:

We evaluate the proposed GCRF network using the challenging PASCALVOC 2012 segmentation dataset [12], which consists of 20 object classes and one background class. The original dataset consists of 1464, 1449 and 1456 training, validation and test images, respectively. Similar to [7, 55], we augment the training set with the additional annotations provided by [16], resulting in a total of 10,582 training images. For quantitative evaluation, we use the standard mean intersection-over-union measure (averaged across the 21 classes).

Parameters: In our GCRF model, each node was connected to every other node along both rows and columns (Figure 2) within a 23×23 spatial neighborhood. Note that since our GCRF model is applied to the CNN output whose resolution is 1/8 times the input resolution, the effective neighborhood size in the input image is 184×184. For our experiments, we used a five layer GMF network, which performs five full-image updates in the forward pass. During training, we used a value of 0.5 for the margin T used in our loss function. The number of filters M used in the similarity layer was set to be equal to the number of classes.

5.1. Training

We used the open source Caffe framework [23] for training and testing our network. We initialized both of our DeepLab CNNs with the trained model provided by the authors of [7]. Note that this model was finetuned using only the PASCALVOC segmentation data starting from ImageNet-trained VGG-16 model. For training, we used stochastic gradient descent with a weight decay of 5×10^{-3} and momentum of 0.9.

Pretraining: Before training the full GCRF network, we pre-trained the similarity layer and CNN of the pairwise

Method	bkg	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbk	person	plant	sheep	sofa	train	tv	mean
MSRA-CFM [9]	87.7	75.7	26.7	69.5	48.8	65.6	81.0	69.2	73.3	30.0	68.7	51.5	69.1	68.1	71.7	67.5	50.4	66.5	44.4	58.9	53.5	61.8
FCN-8s [30]	91.2	76.8	34.2	68.9	49.4	60.3	75.3	74.7	77.6	21.4	62.5	46.8	71.8	63.9	76.5	73.9	45.2	72.4	37.4	70.9	55.1	62.2
Hypercolumns [18]	89.3	68.7	33.5	69.8	51.3	70.2	81.1	71.9	74.9	23.9	60.6	46.9	72.1	68.3	74.5	72.9	52.6	64.4	45.4	64.9	57.4	62.6
DeepLab CNN [7]	91.6	78.7	51.5	75.8	59.5	61.9	82.5	76.6	79.4	26.9	67.7	54.7	74.3	70.0	79.8	77.3	52.6	75.2	46.6	66.9	57.3	67.0
ZoomOut [31]	91.1	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
Deep message passing [27]	93.9	90.1	38.6	77.8	61.3	74.3	89.0	83.4	83.3	36.2	80.2	56.4	81.2	81.4	83.1	82.9	59.2	83.4	54.3	80.6	70.8	73.4
Approaches that use CNNs and discrete CRFs																						
Deep structure models [28]	93.6	86.7	36.9	82.3	63.0	74.2	89.8	84.1	84.1	32.8	65.4	52.1	79.7	72.1	77.6	81.7	55.6	77.4	37.4	81.4	68.4	70.3
DeconvNet + CRF [32]	92.9	87.8	41.9	80.6	63.9	67.3	88.1	78.4	81.3	25.9	73.7	61.2	72.0	77.0	79.9	78.7	59.5	78.3	55.0	75.2	61.5	70.5
object clique potentials [37]	92.8	80.0	53.8	80.8	62.5	64.7	87.0	78.5	83.0	29.0	82.0	60.3	76.3	78.4	83.0	79.8	57.0	80.0	53.1	70.1	63.1	71.2
DeepLab CNN-CRF [7]	93.3	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 [55]	94.0	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 + FCN + CRF [32]	93.1	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
Proposed GCRF network	93.4	85.2	43.9	83.3	65.2	68.3	89.0	82.7	85.3	31.1	79.5	63.3	80.5	79.3	85.5	81.0	60.5	85.5	52.0	77.3	65.1	73.2

Table 1: Comparison with state-of-the-art on PASCALVOC 2012 test set (when trained using ImageNet and PASCALVOC data).

network such that the output s_{ij} of the similarity layer is high for a pair of pixels that have same class label and low for a pair of pixels that have different class labels. For pre-training, we used the following loss function for each pair of connected pixels:

$$L_{ij} = -\mathbb{1}[l_i = l_j] s_{ij} + \mathbb{1}[l_i \neq l_j] \min(0, s_{ij} - h), \quad (7)$$

where l_i and l_j are respectively the class labels of pixel i and j, and h is a threshold parameter. This loss function encourages s_{ij} to be high for similar pairs and below a threshold h for dissimilar pairs. The value of h was chosen as e^{-10}. For training, we used a mini-batch of 15 images and a starting learning rate of 10^{-3} for the similarity layer parameters $\{\mathbf{f}_m\}$ and 10^{-4} for the CNN parameters θ_p^{CNN}. After training for 8000 iterations, we multiplied the learning rate of the similarity layer parameters by 0.1 and trained for additional 5000 iterations.

Finetuning: After the pre-training stage, we finetuned the entire GCRF network using a mini-batch of 5 images and a starting learning rate of 10^{-2} for all parameters except θ_u^{CNN}, for which we used a small learning rate of 10^{-6}. [3] After training for 6000 iterations, we multiplied the learning rate by 0.01 and trained for additional 25000 iterations.

5.2. Results

Table 1 compares the proposed GCRF network with state-of-the-art semantic segmentation approaches on the challenging PASCALVOC 2012 test set. We can infer the following from these results:

- The proposed GCRF network performs significantly (6.2 points) better than the DeepLab CNN, which was used for initializing our unary network. This shows that GCRFs can be successfully used to model output interactions in discrete labeling problems even though they are continuous models.
- The proposed approach outperforms several recent approaches that use discrete CRF models with CNNs. This shows that, despite being a continuous model, GCRF can be a strong competitor to discrete CRFs in discrete labeling tasks.
- Our result is on par with the state-of-the-art (lower by just 0.2 points) when trained using only ImageNet and PASCALVOC data.

Figure 4 provides a visual comparison of the proposed approach with DeepLab CNN (which is same as our unary network) and DeepLab CNN + discrete CRF. As we can see, the proposed GCRF model is able to correct the errors made by the unary network, and also produces more accurate segmentation maps compared to the discrete CRF-based DeepLab approach.

Computation time: The proposed GCRF network takes around 0.6 seconds to segment a 505×505 image on an NVIDIA TITAN GPU.

6. Conclusions

In this work, we proposed to use a GCRF model for the discrete labeling task of semantic segmentation. We proposed a novel deep network, which we refer to as GMF network, by unfolding a fixed number of Gaussian mean field inference steps. By combining this GMF network with CNNs, we proposed an end-to-end trainable GCRF network. When trained discriminatively, the proposed GCRF network outperformed various recent discrete CRF-based semantic segmentation approaches on the challenging PASCALVOC 2012 segmentation dataset. Our results suggest that, despite being a continuous model, GCRF can be successfully used for discrete labeling tasks.

[3]Since the Unary DeepLab CNN was trained by [7] using PASCALVOC segmentation data, it was already close to a good local minima. Hence, we finetuned it with a small learning rate.

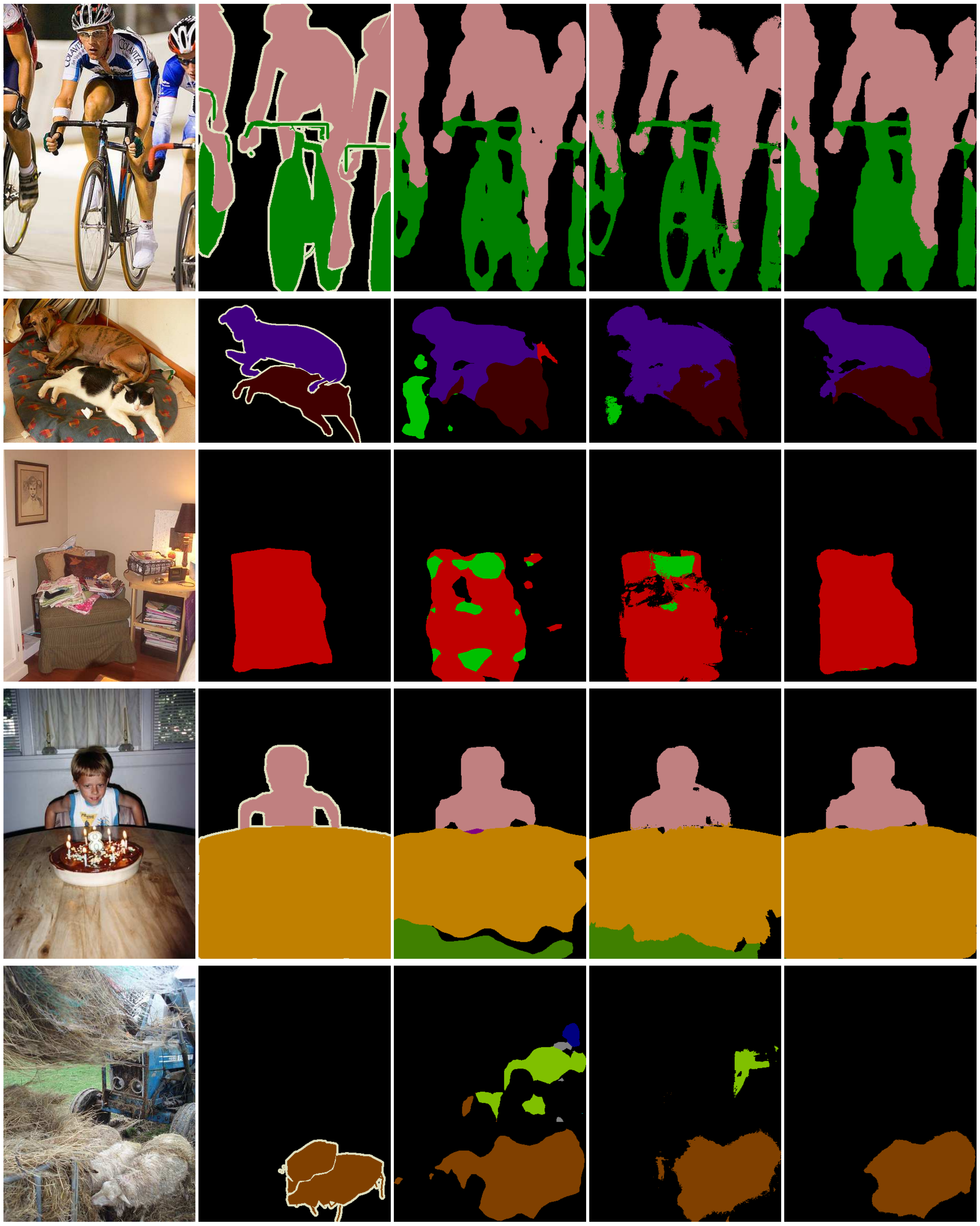

Figure 4: Comparison of the proposed approach with DeepLab CNN [7] and DeepLab CNN + discrete CRF [7].

References

[1] T. Baltrusaitis, P. Robinson, and L. Morency. Continuous Conditional Neural Fields for Structured Regression. In *ECCV*, 2014. 4

[2] A. Barbu. Training an Active Random Field for Real-Time Image Denoising. *IEEE Transactions on Image Processing*, 18(11):2451–2462, 2009. 3

[3] S. Bell, P. Upchurch, N. Snavely, and K. Bala. Material Recognition in the Wild with the Materials in Context Database. In *CVPR*, 2015. 1, 3

[4] Y. Bengio, Y. LeCun, and D. Henderson. Globally Trained Handwritten Word Recognizer Using Spatial Representation, Convolutional Neural Networks, and Hidden Markov Models. In *NIPS*, 1993. 3

[5] R. Chellappa and S. Chatterjee. Classification of Textures Using Gaussian Markov Random Fields. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33(4):959–963, 1985. 1

[6] L. Chen, A. G. Schwing, A. L. Yuille, and R. Urtasun. Learning Deep Structured Models. In *ICML*, 2015. 3

[7] 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 *ICLR*, 2015. 1, 3, 5, 6, 7, 8

[8] J. Dai, K. He, and J. Sun. BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation. In *ICCV*, 2015. 3

[9] J. Dai, K. He, and J. Sun. Convolutional Feature Masking for Joint Object and Stuff Segmentation. In *CVPR*, 2015. 3, 7

[10] T. M. T. Do and T. Artières. Neural Conditional Random Fields. In *AISTATS*, 2010. 3

[11] J. Domke. Learning Graphical Model Parameters with Approximate Marginal Inference. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(10):2454–2467, 2013. 3

[12] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, pages 98–136, 2014. 2, 6

[13] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning Hierarchical Features for Scene Labeling. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1915–1929, 2013. 1, 3

[14] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *CVPR*, 2014. 3

[15] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *ICML*, 2010. 3

[16] B. Hariharan, P. Arbelaez, L. D. Bourdev, S. Maji, and J. Malik. Semantic Contours from Inverse Detectors. In *ICCV*, 2011. 6

[17] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Simultaneous Detection and Segmentation. In *ECCV*, 2014. 3

[18] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Hypercolumns for Object Segmentation and Fine-grained Localization. In *CVPR*, 2015. 1, 3, 5, 7

[19] J. R. Hershey, J. L. Roux, and F. Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *CoRR*, abs/1409.2574, 2014. 3

[20] S. Hong, H. Noh, and B. Han. Decoupled Deep Neural Network for Semi-supervised Semantic Segmentation. In *NIPS*, 2015. 3

[21] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Deep Structured Output Learning for Unconstrained Text Recognition. In *ICLR*, 2014. 3

[22] J. Jancsary, S. Nowozin, and C. Rother. Loss-specific Training of Non-parametric Image Restoration Models: A New State of the Art. In *ECCV*, 2012. 1, 3

[23] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*, 2014. 6

[24] P. Krähenbühl and V. Koltun. Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In *NIPS*, 2011. 2, 3

[25] P. Krähenbühl and V. Koltun. Parameter Learning and Convergent Inference for Dense Random Fields. In *ICML*, pages 513–521, 2013. 3

[26] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, 2001. 1, 3

[27] G. Lin, C. Shen, I. Reid, and A. V. D. Hengel. Deeply Learning the Messages in Message Passing Inference. In *NIPS*, 2015. 3, 7

[28] G. Lin, C. Shen, I. D. Reid, and A. V. D. Hengel. Efficient Piecewise Training of Deep Structured Models for Semantic Segmentation. *CoRR*, abs/1504.01013, 2015. 3, 7

[29] F. Liu, C. Shen, and G. Lin. Deep Convolutional Neural Fields for Depth Estimation from a Single Image. In *CVPR*, 2015. 1, 3, 4

[30] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *CVPR*, 2015. 1, 3, 7

[31] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich. Feedforward Semantic Segmentation with Zoom-out Features. In *CVPR*, 2015. 1, 3, 7

[32] H. Noh, S. Hong, and B. Han. Learning Deconvolution Network for Semantic Segmentation. In *ICCV*, 2015. 3, 7

[33] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly and Semi-Supervised Learning of a DCNN for Semantic Image Segmentation. In *ICCV*, 2015. 1, 3

[34] J. Peng, L. Bo, and J. Xu. Conditional Neural Fields. In *NIPS*, 2009. 3

[35] P. H. O. Pinheiro and R. Collobert. Recurrent Convolutional Neural Networks for Scene Labeling. In *ICML*, 2014. 1

[36] P. O. Pinheiro and R. Collobert. From image-level to pixel-level labeling with Convolutional Networks. In *CVPR*, 2015. 3

[37] X. Qi, J. Shi, S. Liu, R. Liao, and J. Jia. Semantic Segmentation with Object Clique Potentials. In *ICCV*, 2015. 1, 3, 7

[38] V. Radosavljevic, S. Vucetic, and Z. Obradovic. Neural Gaussian Conditional Random Fields. In *ECML*, 2014. 4

[39] S. Ross, D. Munoz, M. Hebert, and J. A. Bagnell. Learning Message-passing Inference Machines for Structured Prediction. In *CVPR*, 2011. 3

[40] S. Roth and M. J. Black. Fields of Experts. *International Journal of Computer Vision*, 82(2):205–229, 2009. 3

[41] H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall, London, 2005. 1

[42] A. Saxena, S. H. Chung, and A. Y. Ng. 3-D Depth Reconstruction from a Single Still Image. *International Journal of Computer Vision*, 76(1):53–69, 2008. 1, 3

[43] U. Schmidt, J. Jancsary, S. Nowozin, S. Roth, and C. Rother. Cascades of Regression Tree Fields for Image Restoration. *IEEE Trans. Pattern Anal. Mach. Intell.*, To appear, 2015. 1, 3

[44] U. Schmidt and S. Roth. Shrinkage Fields for Effective Image Restoration. In *CVPR*, 2014. 3

[45] A. G. Schwing and R. Urtasun. Fully Connected Deep Structured Networks. *CoRR*, abs/1503.02351, 2015. 1, 3

[46] A. Sharma, O. Tuzel, and D. W. Jacobs. Deep Hierarchical Parsing for Semantic Segmentation. In *CVPR*, 2015. 3

[47] A. Sharma, O. Tuzel, and M.-Y. Liu. Recursive Context Propagation Network for Semantic Scene Labeling. In *NIPS*, 2015. 3

[48] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5

[49] V. Stoyanov, A. Ropson, and J. Eisner. Empirical Risk Minimization of Graphical Model Parameters given Approximate Inference, Decoding, and Model Structure. In *AISTATS*, 2011. 3

[50] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. Learning Gaussian Conditional Random Fields for Low-Level Vision. In *CVPR*, 2007. 1, 3

[51] M. F. Tappen, K. G. G. Samuel, C. V. Dean, and D. M. Lyle. The Logistic Random Field - A Convenient Graphical Model for Learning Parameters for MRF-based Labeling. In *CVPR*, 2008. 3

[52] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint Training of a Convolutional Network and a Graphical Model for Human Pose Estimation. In *NIPS*, 2014. 3

[53] M. J. Wainwright and M. I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008. 2, 4

[54] J. Zerubia and R. Chellappa. Mean Field Annealing Using Compound Gauss-Markov Random Fields for Edge Detection and Image Estimation. *IEEE Transactions on Neural Networks*, 4(4):703–709, 1993. 1

[55] S. Zheng, S. Jayasumana, B. R.-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional Random Fields as Recurrent Neural Networks. In *ICCV*, 2015. 1, 3, 6, 7

[56] D. Zoran and Y. Weiss. From Learning Models of Natural Image Patches to Whole Image Restoration. In *ICCV*, 2011. 1, 3