Deep Convolutional Neural Fields for Depth Estimation from a Single Image

Fayao Liu, Chunhua Shen, Guosheng Lin
University of Adelaide, Australia; Australian Centre for Robotic Vision

Abstract

We consider the problem of depth estimation from a single monocular image in this work. It is a challenging task as no reliable depth cues are available, e.g.*, stereo correspondences, motions* etc. *Previous efforts have been focusing on exploiting geometric priors or additional sources of information, with all using hand-crafted features. Recently, there is mounting evidence that features from deep convolutional neural networks (CNN) are setting new records for various vision applications. On the other hand, considering the continuous characteristic of the depth values, depth estimations can be naturally formulated into a continuous conditional random field (CRF) learning problem. Therefore, we in this paper present a deep convolutional neural field model for estimating depths from a single image, aiming to jointly explore the capacity of deep CNN and continuous CRF. Specifically, we propose a deep structured learning scheme which learns the unary and pairwise potentials of continuous CRF in a unified deep CNN framework.*

The proposed method can be used for depth estimations of general scenes with no geometric priors nor any extra information injected. In our case, the integral of the partition function can be analytically calculated, thus we can exactly solve the log-likelihood optimization. Moreover, solving the MAP problem for predicting depths of a new image is highly efficient as closed-form solutions exist. We experimentally demonstrate that the proposed method outperforms state-of-the-art depth estimation methods on both indoor and outdoor scene datasets.

1. Introduction

Estimating depths from a single monocular image depicting general scenes is a fundamental problem in computer vision, which has found wide applications in scene understanding, 3D modelling, robotics, *etc*. It is a notoriously ill-posed problem, as one captured image may correspond to numerous real world scenes [1]. Whereas for humans, inferring the underlying 3D structure from a single image is of little difficulties, it remains a challenging task for computer vision algorithms as no reliable cues can be exploited, such as temporal information, stereo correspondences, *etc*. Previous works mainly focus on enforcing geometric assumptions, *e.g*., box models, to infer the spatial layout of a room [2,3] or outdoor scenes [4]. These models come with innate restrictions, which are limitations to model only particular scene structures and therefore not applicable for general scene depth estimations. Later on, non-parametric methods [5] are explored, which consists of candidate images retrieval, scene alignment and then depth infer using optimizations with smoothness constraints. This is based on the assumption that scenes with semantic similar appearances should have similar depth distributions when densely aligned. However, this method is prone to propagate errors through the different decoupled stages and relies heavily on building a reasonable sized image database to perform the candidates retrieval. In recent years, efforts have been made towards incorporating additional sources of information, *e.g*., user annotations [6], semantic labellings [7,8]. In the recent work of [8], Ladicky *et al*. have shown that jointly performing depth estimation and semantic labelling can benefit each other. Nevertheless, all these methods use hand-crafted features.

Different from the previous efforts, we propose to formulate the depth estimation as a deep continuous CRF learning problem, without relying on any geometric priors nor any extra information. Conditional Random Fields (CRF) [9] are popular graphical models used for structured prediction. While extensively studied in classification (discrete) domains, CRF has been less explored for regression (continuous) problems. One of the pioneering work on continuous CRF can be attributed to [10], in which it was proposed for global ranking in document retrieval. Under certain constraints, they can directly solve the maximum likelihood optimization as the partition function can be analytically calculated. Since then, continuous CRF has been applied for solving various structured regression problems, *e.g*., remote sensing [11,12], image denoising [12]. Motivated by all these successes, we here propose to use it for depth estimation, given the continuous nature of the depth values, and learn the potential functions in a deep convolutional neural network (CNN).

Recent years have witnessed the prosperity of deep con-

volutional neural networks (CNN). CNN features have been setting new records for a wide variety of vision applications [13]. Despite all the successes in classification problems, deep CNN has been less explored for structured learning problems, *i.e*., joint training of a deep CNN and a graphical model, which is a relatively new and not well addressed problem. To our knowledge, no such model has been successfully used for depth estimations. We here bridge this gap by jointly exploring CNN and continuous CRF.

To sum up, we highlight the main contributions of this work as follows:

- We propose a deep convolutional neural field model for depth estimations by exploring CNN and continuous CRF. Given the continuous nature of the depth values, the partition function in the probability density function can be analytically calculated, therefore we can directly solve the log-likelihood optimization without any approximations. The gradients can be exactly calculated in the back propagation training. Moreover, solving the MAP problem for predicting the depth of a new image is highly efficient since closed form solutions exist.
- We jointly learn the unary and pairwise potentials of the CRF in a unified deep CNN framework, which is trained using back propagation.
- We demonstrate that the proposed method outperforms state-of-the-art results of depth estimation on both indoor and outdoor scene datasets.

2. Related work

Prior works [7,14,15] typically formulate the depth estimation as a Markov Random Field (MRF) learning problem. As exact MRF learning and inference are intractable in general, most of these approaches employ approximation methods, *e.g*., multi-conditional learning (MCL), particle belief propagation (PBP). Predicting the depths of a new image is inefficient, taking around 4-5s in [15] and even longer (30s) in [7]. Furthermore, these methods suffer from lacking of flexibility in that [14,15] rely on horizontal alignment of images and [7] requires the semantic labellings of the training data available beforehand. More recently, Liu *et al*. [16] propose a discrete-continuous CRF model to take into consideration the relations between adjacent superpixels, *e.g*., occlusions. They also need to use approximation methods for learning and MAP inference. Besides, their method relies on image retrievals to obtain a reasonable initialization. By contrast, we here present a deep continuous CRF model in which we can directly solve the log-likelihood optimization without any approximations as the partition function can be analytically calculated. Predicting the depth of a new image is highly efficient since a closed form solution exists. Moreover, our model does not inject any geometric priors or any extra information.

On the other hand, previous methods [5,7,8,15,16] all use hand-crafted features in their work, *e.g*., texton, GIST, SIFT, PHOG, object bank, *etc*. In contrast, we learn deep CNN for constructing unary and pairwise potentials of CRF. By jointly exploring the capacity of CNN and continuous CRF, our method outperforms state-of-the-art methods on both indoor and outdoor scene depth estimations. Perhaps the most related work is the recent work of [1], which is concurrent to our work here. They train two CNNs for depth map prediction from a single image. However, our method differs critically from theirs: they directly regress the depth map from an input image through convolutions; in contrast we use a CRF to explicitly model the relations of neighboring superpixels, and learn the potentials (both unary and pairwise) in a unified CNN framework. Moreover, the predicted depth map of [1] is 1/4-resolution of the original input image with some border areas lost, while our method does not have this limitation.

In the most recent work of [17], Tompson *et al*. present a hybrid architecture for jointly training a deep CNN and an MRF for human pose estimation. They first train a unary term and a spatial model separately, then jointly learn them as a fine tuning step. During fine tuning of the whole model, they simply remove the partition function in the likelihood to have a loose approximation. In contrast, our model performs continuous variables prediction. We can directly solve the log-likelihood optimization without using approximations as the partition function is integrable and can be analytically calculated. Moreover, during prediction, we have a closed-form solution for the MAP inference. Although no convolutions are involved, the work of [18] shares similarity with ours in that both use neural networks to model the potentials of continuous CRF. Note that the model in [18] only consists of one (fully connected) hidden layer, while ours uses deep CNNs. It is unclear how the method of [18] performs on the challenging depth estimation problem that we consider here.

3. Deep convolutional neural fields

We present the details of our deep convolutional neural field model for depth estimation in this section. Unless otherwise stated, we use boldfaced uppercase and lowercase letters to denote matrices and column vectors respectively.

3.1. Overview

The goal here is to infer the depth of each pixel in a single image depicting general scenes. Following the work of [7,15,16], we make the common assumption that an image is composed of small homogeneous regions (superpixels) and consider the graphical model composed of nodes defined on superpixels. Note that our framework is flexi-

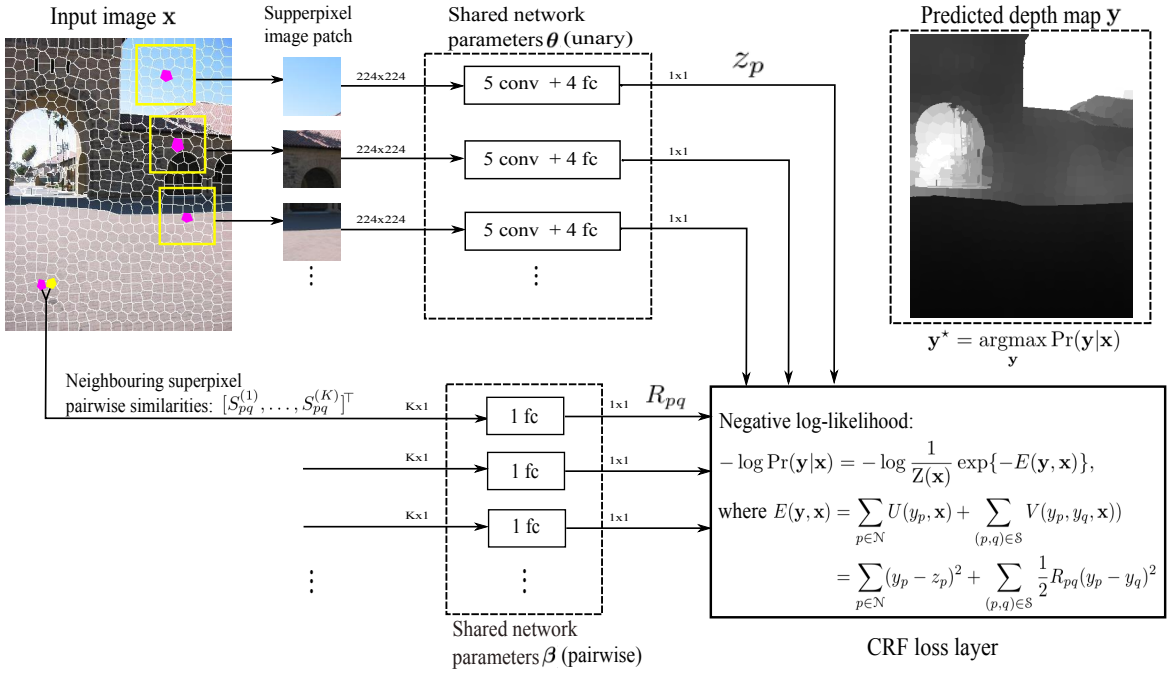

Figure 1: An illustration of our deep convolutional neural field model for depth estimation. The input image is first over-segmented into superpixels. In the unary part, for a superpixel p, we crop the image patch centred around its centroid, then resize and feed it to a CNN which is composed of 5 convolutional and 4 fully-connected layers (details refer to Fig. 2). In the pairwise part, for a pair of neighbouring superpixels (p, q), we consider K types of similarities, and feed them into a fully-connected layer. The outputs of unary part and the pairwise part are then fed to the CRF structured loss layer, which minimizes the negative log-likelihood. Predicting the depths of a new image $\mathbf{x}$ is to maximize the conditional probability $\Pr(\mathbf{y}|\mathbf{x})$, which has closed-form solutions (see Sec. 3.3 for details).

ble that can work on pixels or superpixels. Each superpixel is portrayed by the depth of its centroid. Let $\mathbf{x}$ be an image and $\mathbf{y} = [y_1, \ldots, y_n]^\top \in \mathbb{R}^n$ be a vector of continuous depth values corresponding to all n superpixels in $\mathbf{x}$. Similar to conventional CRF, we model the conditional probability distribution of the data with the following density function:

$$\Pr(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathrm{Z}(\mathbf{x})} \exp(-E(\mathbf{y},\mathbf{x})), \tag{1}$$

where E is the energy function; Z is the partition function defined as:

$$\mathrm{Z}(\mathbf{x}) = \int_{\mathbf{y}} \exp\{-E(\mathbf{y},\mathbf{x})\} \mathrm{d}\mathbf{y}. \tag{2}$$

Here, because $\mathbf{y}$ is continuous, the integral in Eq. (1) can be analytically calculated under certain circumstances, which we will show in Sec. 3.3. This is different from the discrete case, in which approximation methods need to be applied. To predict the depths of a new image, we solve the maximum a posteriori (MAP) inference problem:

$$\mathbf{y}^\star = \operatorname*{argmax}_{\mathbf{y}} \Pr(\mathbf{y}|\mathbf{x}). \tag{3}$$

We formulate the energy function as a typical combination of unary potentials U and pairwise potentials V over the nodes (superpixels) $\mathcal{N}$ and edges $\mathcal{S}$ of the image $\mathbf{x}$:

$$E(\mathbf{y},\mathbf{x}) = \sum_{p\in\mathcal{N}} U(y_p,\mathbf{x}) + \sum_{(p,q)\in\mathcal{S}} V(y_p,y_q,\mathbf{x}). \tag{4}$$

The unary term U aims to regress the depth value from a single superpixel. The pairwise term V encourages neighbouring superpixels with similar appearances to take similar depths. We aim to jointly learn U and V in a unified CNN framework.

In Fig. 1, we show a sketch of our deep convolutional neural field model for depth estimation. As we can see, the whole network is composed of a unary part, a pairwise part and a CRF loss layer. For an input image, which has been over-segmented into n superpixels, we consider image patches centred around each superpxiel centroid. The unary part then takes all the image patches as input and feed each of them to a CNN and output an n-dimentional vector containing regressed depth values of the n superpixels. The network for the unary part is composed of 5 convolutional and 4 fully-connected layers with details in Fig. 2. Note that the CNN parameters are shared across all the superpixels. The pairwise part takes similarity vectors (each with K com-

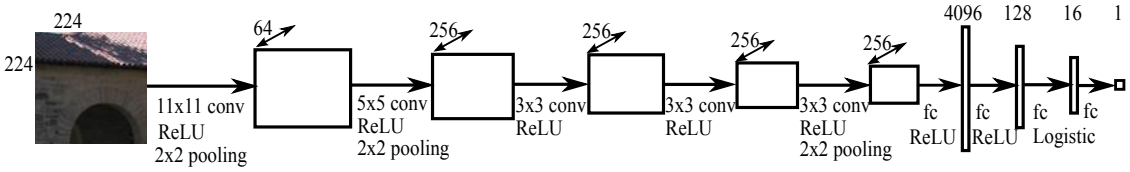

Figure 2: Detailed network architecture of the unary part in Fig. 1.

ponents) of all neighbouring superpixel pairs as input and feed each of them to a fully-connected layer (parameters are shared among different pairs), then output a vector containing all the 1-dimentional similarities for each of the neighbouring superpixel pair. The CRF loss layer takes as input the outputs from the unary and the pairwise parts to minimize the negative log-likelihood. Compared to the direct regression method in [1], our model possesses two potential advantages: 1) We achieve translation invariance as we construct unary potentials irrespective of the superpixel's coordinate (shown in Sec. 3.2); 2) We explicitly model the relations of neighbouring superpixels through pairwise potentials.

In the following, we describe the details of potential functions involved in the energy function in Eq. (4).

3.2. Potential functions

Unary potential The unary potential is constructed from the output of a CNN by considering the least square loss:

$$U(y_p, \mathbf{x}; \boldsymbol{\theta}) = (y_p - z_p(\boldsymbol{\theta}))^2, \quad \forall p = 1, ..., n. \tag{5}$$

Here z_p is the regressed depth of the superpixel p parametrized by the CNN parameters $\boldsymbol{\theta}$.

The network architecture for the unary part is depicted in Fig. 2. Our CNN model in Fig. 2 is mainly based upon the well-known network architecture of Krizhevsky *et al.* [19] with modifications. It is composed of 5 convolutional layers and 4 fully connected layers. The input image is first oversegmented into superpixels, then for each superpixel, we consider the image patch centred around its centroid. Each of the image patches is resized to 224×224 pixels and then fed to the convolutional neural network. Note that the convolutional and the fully-connected layers are shared across all the image patches of different superpixels. Rectified linear units (ReLU) are used as activiation functions for the five convolutional layers and the first two fully connected layers. For the third fully-connected layer, we use the logistic function ($f(x) = (1 + e^{-x})^{-1}$) as activiation function. The last fully-connected layer has no activiation function followed. The output is an 1-dimentional real-valued depth for a single superpixel.

Pairwise potential We construct the pairwise potential from K types of similarity observations, each of which enforces smoothness by exploiting consistency information of neighbouring superpixels:

$$V(y_p, y_q, \mathbf{x}; \boldsymbol{\beta}) = \frac{1}{2} R_{pq} (y_p - y_q)^2, \ \forall p, q = 1, ..., n. \tag{6}$$

Here R_{pq} is the output of the network in the pairwise part (see Fig. 1) from a neighbouring superpixel pair (p, q). We use a fully-connected layer here:

$$R_{pq} = \boldsymbol{\beta}^\top [S_{pq}^{(1)}, \ldots, S_{pq}^{(K)}]^\top = \sum_{k=1}^{K} \beta_k S_{pq}^{(k)}, \tag{7}$$

where $\mathbf{S}^{(k)}$ is the k-th similarity matrix whose elements are $S_{pq}^{(k)}$ ($\mathbf{S}^{(k)}$ is symmetric); $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_k]^\top$ are the network parameters. From Eq. (7), we can see that we don't use any activiation function. However, as our framework is general, more complicated networks can be seamlessly incorporated for the pairwise part. In Sec .3.3, we will show that we can derive a general form for calculating the gradients with respect to $\boldsymbol{\beta}$ (see Eq. (16)). To guarantee $Z(x)$ (Eq. (2)) is integrable, we require $\beta_k \geq 0$ [10].

We consider 3 types of pairwise similarities, measured by the color difference, color histogram difference and texture disparity in terms of local binary patterns (LBP) [20], which take the conventional form: $S_{pq}^{(k)} = e^{-\gamma \| s_p^{(k)} - s_q^{(k)} \|}, k = 1, 2, 3$, where $s_p^{(k)}$, $s_q^{(k)}$ are the observation values of the superpixel p, q calculated from color, color histogram and LBP; $\|\cdot\|$ denotes the ℓ_2 norm of a vector and γ is a constant.

3.3. Learning

With the unary and the pairwise pontentials defined in Eq. (5), (6), we can now write the energy function as:

$$E(\mathbf{y}, \mathbf{x}) = \sum_{p \in \mathcal{N}} (y_p - z_p)^2 + \sum_{(p,q) \in \mathcal{S}} \frac{1}{2} R_{pq} (y_p - y_q)^2. \tag{8}$$

For ease of expression, we introduce the following notation:

$$\mathbf{A} = \mathbf{I} + \mathbf{D} - \mathbf{R}, \tag{9}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix; $\mathbf{R}$ is the matrix composed of R_{pq}; $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{pp} = \sum_q R_{pq}$. Expanding Eq. (8), we have:

$$E(\mathbf{y}, \mathbf{x}) = \mathbf{y}^\top \mathbf{A} \mathbf{y} - 2\mathbf{z}^\top \mathbf{y} + \mathbf{z}^\top \mathbf{z}. \tag{10}$$

Due to the quadratic terms of $\mathbf{y}$ in the energy function in Eq. (10) and the positive definiteness of $\mathbf{A}$, we can analytically calculate the integral in the partition function (Eq. (2)) as:

$$\begin{aligned} Z(\mathbf{x}) &= \int_{\mathbf{y}} \exp\{-E(\mathbf{y},\mathbf{x})\}\mathrm{d}\mathbf{y} \\ &= \frac{(\pi)^{\frac{n}{2}}}{|\mathbf{A}|^{\frac{1}{2}}} \exp\{\mathbf{z}^\top \mathbf{A}^{-1}\mathbf{z} - \mathbf{z}^\top \mathbf{z}\}. \end{aligned} \tag{11}$$

From Eq. (1), (10), (11), we can now write the probability distribution function as:

$$\Pr(\mathbf{y}|\mathbf{x}) = \frac{|\mathbf{A}|^{\frac{1}{2}}}{\pi^{\frac{n}{2}}} \exp\Big\{ -\mathbf{y}^\top \mathbf{A}\mathbf{y} + 2\mathbf{z}^\top \mathbf{y} - \mathbf{z}^\top \mathbf{A}^{-1}\mathbf{z}\Big\}, \tag{12}$$

where $\mathbf{z} = [z_1, \ldots, z_n]^\top$; $|\mathbf{A}|$ denotes the determinant of the matrix $\mathbf{A}$, and $\mathbf{A}^{-1}$ the inverse of $\mathbf{A}$. Then the negative log-likelihood can be written as:

$$\begin{aligned} -\log\Pr(\mathbf{y}|\mathbf{x}) = \mathbf{y}^\top \mathbf{A}\mathbf{y} - 2\mathbf{z}^\top \mathbf{y} + \mathbf{z}^\top \mathbf{A}^{-1}\mathbf{z} \\ - \frac{1}{2}\log(|\mathbf{A}|) + \frac{n}{2}\log(\pi). \end{aligned} \tag{13}$$

During learning, we minimizes the negative conditional log-likelihood of the training data. Adding regularization to $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, we then arrive at the final optimization:

$$\begin{aligned} \min_{\boldsymbol{\theta},\boldsymbol{\beta}\geq\mathbf{0}} -\sum_{i=1}^{N} \log\Pr(\mathbf{y}^{(i)}|\mathbf{x}^{(i)};\boldsymbol{\theta},\boldsymbol{\beta}) \\ + \frac{\lambda_1}{2}\|\boldsymbol{\theta}\|_2^2 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2, \end{aligned} \tag{14}$$

where $\mathbf{x}^{(i)}$, $\mathbf{y}^{(i)}$ denote the i-th training image and the corresponding depth map; N is the number of training images; λ_1 and λ_2 are weight decay parameters. We use stochastic gradient descent (SGD) based back propagation to solve the optimization problem in Eq. (14) for learning all parameters of the whole network. We project the solutions to the feasible set when the bounded constraints $\beta_k \geq 0$ is violated. In the following, we calculate the partial derivatives of $-\log\Pr(\mathbf{y}|\mathbf{x})$ with respect to the network parameters θ_l (one element of $\boldsymbol{\theta}$) and β_k (one element of $\boldsymbol{\beta}$) by using the chain rule:

$$\frac{\partial\{-\log\Pr(\mathbf{y}|\mathbf{x})\}}{\partial\theta_l} = 2(\mathbf{A}^{-1}\mathbf{z} - \mathbf{y})^\top \frac{\partial\mathbf{z}}{\partial\theta_l}, \tag{15}$$

$$\begin{aligned} \frac{\partial\{-\log\Pr(\mathbf{y}|\mathbf{x})\}}{\partial\beta_k} = \mathbf{y}^\top \mathbf{J}\mathbf{y} - \mathbf{z}^\top \mathbf{A}^{-1}\mathbf{J}\mathbf{A}^{-1}\mathbf{z} \\ - \frac{1}{2}\mathrm{Tr}\left(\mathbf{A}^{-1}\mathbf{J}\right), \end{aligned} \tag{16}$$

where $\mathrm{Tr}(\cdot)$ denotes the trace of a matrix; $\mathbf{J}$ is an $n \times n$ matrix with elements:

$$J_{pq} = -\frac{\partial R_{pq}}{\partial\beta_k} + \delta(p=q)\sum_q \frac{\partial R_{pq}}{\partial\beta_k}, \tag{17}$$

where $\delta(\cdot)$ is the indicator function, which equals 1 if $p = q$ is true and 0 otherwise. From Eq. (17), we can see that our framework is general and more complicated networks for the pairwise part can be seamlessly incorporated. Here, in our case, with the definition of R_{pq} in Eq. (7), we have $\frac{\partial R_{pq}}{\partial\beta_k} = S_{pq}^{(k)}$.

Depth prediction Predicting the depths of a new image is to solve the MAP inference in Eq. (3), in which closed form solutions exist here:

$$\begin{aligned} \mathbf{y}^\star &= \operatorname*{argmax}_{\mathbf{y}} \Pr(\mathbf{y}|\mathbf{x}) \\ &= \operatorname*{argmax}_{\mathbf{y}} -\mathbf{y}^\top \mathbf{A}\mathbf{y} + 2\mathbf{z}^\top \mathbf{y} \\ &= \mathbf{A}^{-1}\mathbf{z}. \end{aligned} \tag{18}$$

If we discard the pairwise terms, namely $R_{pq} = 0$, then Eq. (18) degenerates to $\mathbf{y}^\star = \mathbf{z}$, which is a conventional regression model (we will report the results of this method as a baseline in the experiment).

3.4. Implementation details

We implement the network training based on the efficient CNN toolbox: VLFeat MatConvNet[1] [21]. Training is done on a standard desktop with an NVIDIA GTX 780 GPU with 6GB memory. During each SGD iteration, around $\sim$ 700 superpixel image patches need to be processed. The 6GB GPU may not be able to process all the image patches at one time. We therefore partition the superpixel image patches of one image into two parts and process them successively. Processing one image takes around 10s (including forward and backward) with $\sim$ 700 superpixels when training the whole network.

During implementation, we initialize the first 6 layers of the unary part in Fig. 2 using a CNN model trained on the ImageNet from [22]. First, we do not back propagate through the previous 6 layers by keeping them fixed and train the rest of the network (we refer this process as pre-train) with the following settings: momentum is set to 0.9, and weight decay parameters λ_1, λ_2 are set to 0.0005. During pre-train, the learning rate is initialized at 0.0001, and decreased by 40% every 20 epoches. We then run 60 epoches to report the results of pre-train (with learning rate decreased twice). The pre-train is rather efficient, taking around 1 hour to train on the Make3D dataset. Then we train the whole network with the same momentum and weight decay. We apply dropout with ratio 0.5 in the first two fully-connected layers of Fig. 2. Training the whole network takes around 16.5 hours on the Make3D dataset, and around 33 hours on the NYU v2 dataset. When predicting the depths of a new image, it takes $\sim$ 1.1s to perform the network forward pass.

[1] VLFeat MatConvNet: http://www.vlfeat.org/matconvnet/

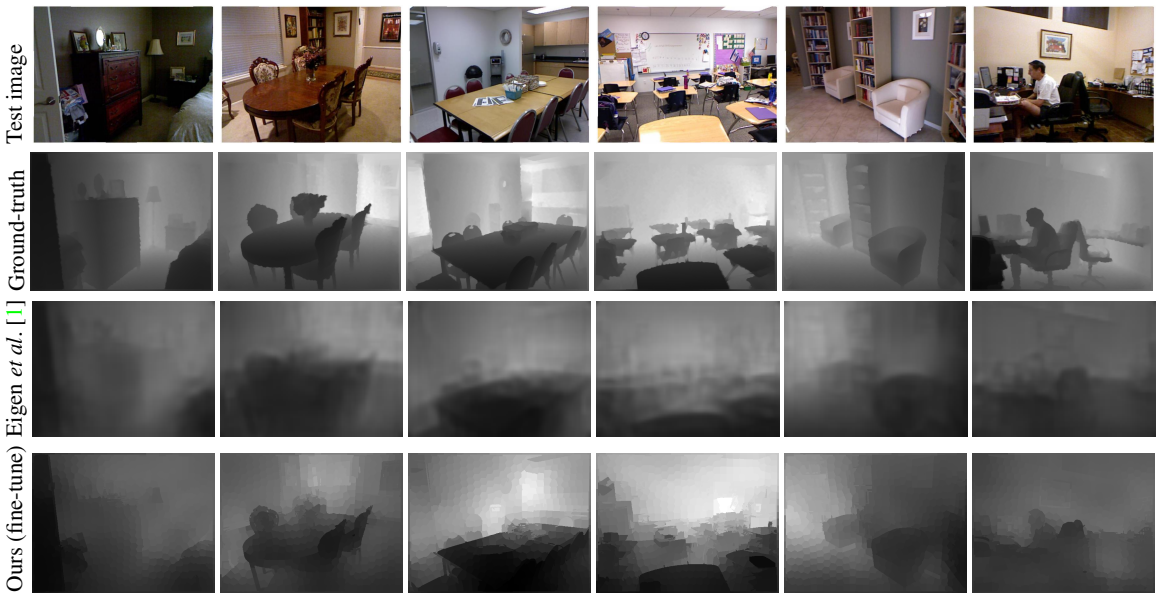

Figure 3: Examples of qualitative comparisons on the NYUD2 dataset (Best viewed on screen). Our method yields visually better predictions with sharper transitions, aligning to local details.

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Make3d [15]	0.349	-	1.214	0.447	0.745	0.897
DepthTransfer [5]	0.35	0.131	1.2	-	-	-
Discrete-continuous CRF [16]	0.335	0.127	1.06	-	-	-
Ladicky *et al.* [8]	-	-	-	0.542	0.829	0.941
Eigen *et al.* [1]	**0.215**	-	0.907	0.611	**0.887**	**0.971**
Ours (pre-train)	0.257	0.101	0.843	0.588	0.868	0.961
Ours (fine-tune)	0.230	**0.095**	**0.824**	**0.614**	0.883	**0.971**

Table 1: Result comparisons on the NYU v2 dataset. Our method performs the best in most cases. Note that the results of Eigen *et al.* [1] are obtained by using extra training data (in the millions in total) while ours are obtained using the standard training set.

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
SVR	0.313	0.128	1.068	0.490	0.787	0.921
SVR (smooth)	0.290	0.116	0.993	0.514	0.821	0.943
Unary only	0.295	0.117	0.985	0.516	0.815	0.938
Unary only (smooth)	0.287	0.112	0.956	0.535	0.828	0.943
Ours (pre-train)	0.257	0.101	0.843	0.588	0.868	0.961
Ours (fine-tune)	**0.230**	**0.095**	**0.824**	**0.614**	**0.883**	**0.971**

Table 2: Baseline comparisons on the NYU v2 dataset. Our method with the whole network training performs the best.

Method	Error (C1) (lower is better)			Error (C2) (lower is better)		
	rel	log10	rms	rel	log10	rms
SVR	0.433	0.158	8.93	0.429	0.170	15.29
SVR (smooth)	0.380	0.140	**8.12**	0.384	0.155	15.10
Unary only	0.366	0.137	8.63	0.363	0.148	14.41
Unary only (smooth)	0.341	0.131	8.49	0.349	0.144	14.37
Ours (pre-train)	0.331	0.127	8.82	0.324	0.134	13.29
Ours (fine-tune)	**0.314**	**0.119**	8.60	**0.307**	**0.125**	**12.89**

Table 3: Baseline comparisons on the Make3D dataset. Our method with the whole network training performs the best.

4. Experiments

We evaluate on two popular datasets which are available online: the NYU v2 Kinect dataset [23] and the Make3D range image dataset [15]. Several measures commonly used in prior works are applied here for quantitative evaluations:

- average relative error (rel): $\frac{1}{T}\sum_p \frac{|d_p^{gt} - d_p|}{d_p^{gt}}$;
- root mean squared error (rms): $\sqrt{\frac{1}{T}\sum_p (d_p^{gt} - d_p)^2}$;
- average $\log_{10}$ error (log10): $\frac{1}{T}\sum_p |\log_{10} d_p^{gt} - \log_{10} d_p|$;
- accuracy with threshold thr: percentage (%) of d_p s.t. $\max(\frac{d_p^{gt}}{d_p}, \frac{d_p}{d_p^{gt}}) = \delta < thr$;

where d_p^{gt} and d_p are the ground-truth and predicted depths respectively at pixel indexed by p, and T is the total number of pixels in all the evaluated images.

Method	Error (C1) (lower is better)			Error (C2) (lower is better)		
	rel	log10	rms	rel	log10	rms
Make3d [15]	-	-	-	0.370	0.187	-
Semantic Labelling [7]	-	-	-	0.379	0.148	-
DepthTransfer [5]	0.355	0.127	9.20	0.361	0.148	15.10
Discrete-continuous CRF [16]	0.335	0.137	9.49	0.338	0.134	**12.60**
Ours (pre-train)	0.331	0.127	8.82	0.324	0.134	13.29
Ours (fine-tune)	**0.314**	**0.119**	**8.60**	**0.307**	**0.125**	12.89

Table 4: Result comparisons on the Make3D dataset. Our method performs the best. Note that the C2 errors of the Discrete-continuous CRF [16] are reported with an ad-hoc post-processing step (train a classifier to label sky pixels and set the corresponding regions to the maximum depth).

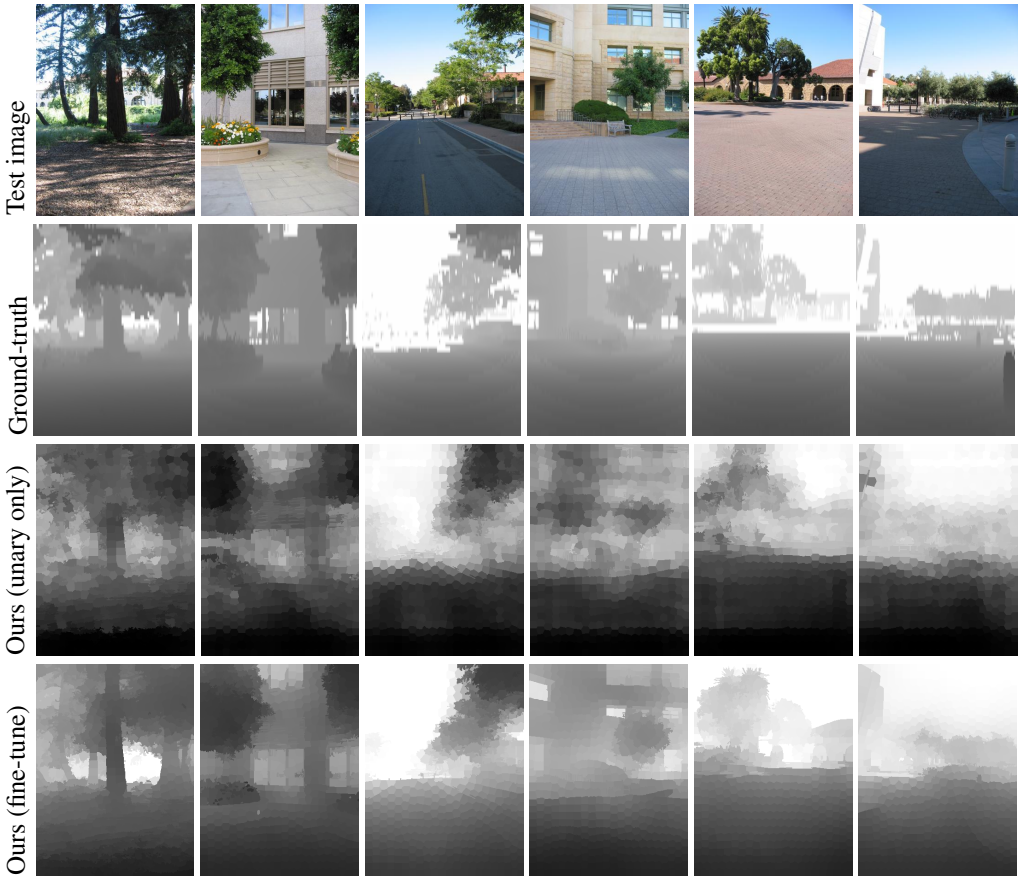

Figure 4: Examples of depth predictions on the Make3D dataset (Best viewed on screen). The unary only model gives rather coarse predictions, with blurry boundaries and segments. In contrast, our full model with pairwise smoothness yields much better predictions.

We use SLIC [24] to segment the images into a set of non-overlapping superpixels. For each superpixel, we consider the image within a rectangular box centred on the centroid of the superpixel, which contains a large portion of its background surroundings. More specifically, we use a box size of 168×168 pixels for the NYU v2 and 120×120 pixels for the Make3D dataset. Following [1,7,15], we transform the depths into log-scale before training. As for baseline comparisons, we consider the following settings:

- SVR: We train a support vector regressor using the CNN representations from the first 6 layers of Fig. 2;
- SVR (smooth): We add a smoothness term to the trained SVR during prediction by solving the inference problem in Eq. (18). As tuning multiple pairwise parameters is not straightforward, we only use color difference as the pairwise potential and choose the parameter β by hand-tuning on a validation set;
- Unary only: We replace the CRF loss layer in Fig. 1

with a least-square regression layer (by setting the pairwise outputs $R_{pq} = 0, p, q = 1, ..., n$), which degenerates to a deep regression model trained by SGD;

- Unary only (smooth): As in the SVR (smooth) model, we add a smoothness term to the trained unary only model during prediction by solving the inference problem in Eq. (18).

4.1. NYU v2: Indoor scene reconstruction

The NYU v2 dataset consists of 1449 RGBD images of indoor scenes, among which 795 are used for training and 654 for test (we use the standard training/test split provided with the dataset). Following [16], we resize the images to 427×561 pixels before training.

For a detailed analysis of our model, we first compare with the three baseline methods and report the results in Table 2. From the table, several conclusions can be made: 1) When trained with only unary term, deeper network is beneficial for better performance, which is demonstrated by the fact that our unary only model outperforms the SVR model; 2) Adding smoothness term to the SVR or our unary only model helps improve the prediction accuracy; 3) Our method achieves the best performance by jointly learning the unary and the pairwise parameters in a unified deep CNN framework. Moreover, fine-tuning the whole network yields further performance gain. These well demonstrate the efficacy of our model.

In Table 1, we compare our model with several popular state-of-the-art methods. As can be observed, our method outperforms classic methods like Make3d [15], DepthTransfer [5] with large margins. Most notably, our results are significantly better than that of [8], which jointly exploits depth estimation and semantic labelling. Comparing to the recent work of Eigen *et al*. [1], our method generally performs on par. Our method obtains significantly better result in terms of root mean square (rms) error. Note that, in [1], they need to collect millions of additional labelled images to train their model. In contrast, we only use the standard training sets (795) without any extra data, yet we achieve comparable or even better performance. Fig. 3 illustrates some qualitative evaluations of our method compared against Eigen *et al*. [1] (We download the predictions of [1] from the authors' website.). Compared to the predictions of [1], our method yields more visually pleasant predictions with sharper transitions, aligning to local details.

4.2. Make3D: Outdoor scene reconstruction

The Make3D dataset contains 534 images depicting outdoor scenes. As pointed out in [15,16], this dataset is with limitations: the maximum value of depths is 81m with faraway objects are all mapped to the one distance of 81 meters. As a remedy, two criteria are used in [16] to report the prediction errors: (C_1) Errors are calculated only in the regions with the ground-truth depth less than 70 meters; (C_2) Errors are calculated over the entire image. We follow this protocol to report the evaluation results.

Likewise, we first present the baseline comparisons in Table 3, from which similar conclusions can be drawn as in the NYU v2 dataset. We then show the detailed results compared with several state-of-the-art methods in Table 4. As can be observed, our model with the whole network training ranks the first in overall performance, outperforming the compared methods by large margins. Note that the C2 errors of [16] are reported with an ad-hoc post-processing step, which trains a classifier to label sky pixels and set the corresponding regions to the maximum depth. In contrast, we do not employ any of those heuristics to refine our results, yet we achieve better results in terms of relative error. Some examples of qualitative evaluations are shown in Fig. 4. It is shown that our unary only model gives rather coarse predictions with blurry boundaries. By adding smoothness term, our model yields much better visualizations, which are close to the ground-truth.

5. Conclusion

We have presented a deep convolutional neural field model for depth estimation from a single image. The proposed method combines the strength of deep CNN and continuous CRF in a unified CNN framework. We show that the log-likelihood optimization in our method can be directly solved using back propagation without any approximations required. Predicting the depths of a new image by solving the MAP inference can be efficiently performed as closed-form solutions exist. Given the general learning framework of our method, it can also be applied for other vision applications, *e.g*., image denoising. Experimental results demonstrate that the proposed method outperforms state-of-the-art methods on both indoor and outdoor scene datasets.

Acknowledgements This work was in part supported by ARC Grant FT120100969; and the Data to Decisions Cooperative Research Centre, Australia.

References

[1] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014. 1, 2, 4, 6, 7, 8

[2] V. Hedau, D. Hoiem, and D. A. Forsyth, "Thinking inside the box: Using appearance models and context based on room geometry," in *Proc. Eur. Conf. Comp. Vis.*, 2010. 1

[3] D. C. Lee, A. Gupta, M. Hebert, and T. Kanade, "Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010. 1

[4] A. Gupta, A. A. Efros, and M. Hebert, "Blocks world re-

visited: Image understanding using qualitative geometry and mechanics," in *Proc. Eur. Conf. Comp. Vis.*, 2010. 1

[5] K. Karsch, C. Liu, and S. B. Kang, "Depthtransfer: Depth extraction from video using non-parametric sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2014. 1, 2, 6, 7, 8

[6] B. C. Russell and A. Torralba, "Building a database of 3d scenes from user annotations," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2009. 1

[7] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2010. 1, 2, 7

[8] L. Ladick, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2014. 1, 2, 6, 8

[9] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. Int. Conf. Mach. Learn.*, 2001. 1

[10] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, and H. Li, "Global ranking using continuous conditional random fields," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008. 1, 4

[11] V. Radosavljevic, S. Vucetic, and Z. Obradovic, "Continuous conditional random fields for regression in remote sensing," in *Proc. Eur. Conf. Artificial Intell.*, 2010. 1

[12] K. Ristovski, V. Radosavljevic, S. Vucetic, and Z. Obradovic, "Continuous conditional random fields for efficient regression in large fully connected graphs," in *AAAI Conf. Artificial Intell.*, 2013. 1

[13] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. Workshop of IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2014. 2

[14] A. Saxena, S. H. Chung, and A. Y. Ng, "Learning depth from single monocular images," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005. 2

[15] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2009. 2, 6, 7, 8

[16] M. Liu, M. Salzmann, and X. He, "Discrete-continuous depth estimation from a single image," in *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2014. 2, 6, 7, 8

[17] 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 Inf. Process. Syst.*, 2014. 2

[18] T. Baltrusaitis, P. Robinson, and L. Morency, "Continuous conditional neural fields for structured regression," in *Proc. Eur. Conf. Comp. Vis.*, 2014. 2

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012. 4

[20] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *Proc. IEEE Int. Conf. Patt. Recogn.*, 1994. 4

[21] A. Vedaldi, "MatConvNet," http://www.vlfeat.org/matconvnet/, 2013. 5

[22] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. British Machine Vis. Conf.*, 2014. 5

[23] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Proc. Eur. Conf. Comp. Vis.*, 2012. 6

[24] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012. 7