

Supplementary Material for Pixel-Adaptive Convolutional Neural Networks

Hang Su¹, Varun Jampani², Deqing Sun², Orazio Gallo², Erik Learned-Miller¹, and Jan Kautz²

¹UMass Amherst ²NVIDIA

In this supplementary document, we provide additional details and results on the deep joint upsampling experiments (Sec. 1) and PAC-CRF (Sec. 2).

1. Deep Joint Upsampling with PAC

Network architecture Here we provide details of our network architectures used in the joint upsampling experiments. Our networks have three branches: Encoder, Guidance, and Decoder. The layers in each branch of the joint depth upsampling networks are listed in Tab. 1. Since we use each PAC^T for 2×, 4×, 8×, 16× networks requires 2, 3, 4 PAC^T layers respectively. The final output from the guidance branch is equally divided in the channel dimension for use as adapting features for the PAC^T layers in the decoder. All CONV and PAC^T layers use 5 × 5 filters, and are followed by ReLU except for the last CONV. We use Gaussian kernels for K in all PAC^T layers.

We design two variants of our model, *standard* and *lite*. The *standard* variant has a simpler design, but has varying number of parameters for different upsampling factors, and overall consume more memory than DJF [2], a previous state-of-the-art approach on joint depth upsampling. For the *lite* variant, we reduce the number of filters and make sure the networks roughly match the number of parameters compared to DJF.

Similar network architectures are also used for optical flow upsampling. First layer of encoder and last layer in decoder are modified to fit the two (u, v) channels in optical flow instead of one channel in depth maps, i.e., using “C2” instead of “C1” in Tab. 1.

Additional examples We provide more joint upsampling visual results for depth (Fig. 1) and optical flow (Fig. 2).

2. Conditional Random Fields

Interpretations of the formulation The pairwise potentials in Full-CRF is defined as $\psi_p(l_i, l_j | I) = \mu(l_i, l_j)K(\mathbf{f}_i, \mathbf{f}_j)$, where the kernel function K has two terms, *appearance kernel* and *smoothness kernel*:

	standard			lite		
	4×	8×	16×	4×	8×	16×
Encoder	C32	C32	C32	C12	C12	C8
	C32	C32	C32	C16	C16	C16
	C32	C32	C32	C22	C16	C16
Guidance	C32	C32	C32	C12	C12	C8
	C32	C32	C32	C22	C16	C16
	C32	C48	C64	C24	C36	C40
Decoder	P32	P32	P32	P12	P12	P8
	P32	P32	P32	P16	P16	P16
	C32	P32	P32	C22	P16	P16
	C1	C32	P32	C1	C20	P16
		C1	C32		C1	C16
#Params	183K	222K	260K	56K	56K	56K

Table 1: **Network architectures for joint depth upsampling.** “C” stands for regular CONV, “P” stands for PAC^T (the transposed convolution variant of PAC), and the number after them represents the number of output channels.

$$\begin{aligned}
 K(\mathbf{f}_i, \mathbf{f}_j) = & w_1 \underbrace{\exp\left\{-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\theta_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\theta_\beta^2}\right\}}_{\text{appearance kernel}} \\
 & + w_2 \underbrace{\exp\left\{-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{2\theta_\gamma^2}\right\}}_{\text{smoothness kernel}}. \quad (1)
 \end{aligned}$$

In comparison, our pairwise potential uses (assuming using Gaussian kernel and a single pairwise term):

$$\begin{aligned}
 K'(\mathbf{f}_i, \mathbf{f}_j) = & \mathbf{W}[\mathbf{p}_j - \mathbf{p}_i]K(\mathbf{f}_i, \mathbf{f}_j) \\
 = & \mathbf{W}[\mathbf{p}_j - \mathbf{p}_i] \exp\left\{-\frac{1}{2}\|\mathbf{f}_i - \mathbf{f}_j\|^2\right\}. \quad (2)
 \end{aligned}$$

There are two major differences:

1. The *smoothness kernel* is now moved out of K and is represented using filter \mathbf{W} . It can still be initialized as a Gaussian, but arbitrary filter is allowed to be learned.
2. The *appearance kernel* now operates on \mathbf{f} directly without the need of decomposing it into multiple parts, and without the individual scaling factors (θ_α, \dots) .

Both changes give the pairwise potential more learning capacity. Note that \mathbf{f} can be the output of some other network layers. A simple linear layer can learn appropriate scaling factors, while in other cases a more complex network may be preferred. For input with more than RGB channels (e.g., 3D data with color, depth, normal, curvature, *etc.*), hand-crafting and finding parameters for kernel functions like Eq. 1 can be time-consuming and suboptimal, and allowing the function to be learned from data in an end-to-end fashion is particularly desirable.

Note that in Eq. 2, \mathbf{W} is a 2D matrix, and the corresponding pairwise potential is defined as

$$\psi_p(l_i, l_j) = \mu(l_i, l_j) \mathbf{W}[\mathbf{p}_j - \mathbf{p}_i] K(\mathbf{f}_i, \mathbf{f}_j), \quad (3)$$

where $\mu(l_i, l_j)$ is the compatibility matrix. Our final pairwise potential, $\psi_p(l_i, l_j) = K(\mathbf{f}_i, \mathbf{f}_j) \mathbf{W}_{l_j l_i}[\mathbf{p}_j - \mathbf{p}_i]$, can be seen as a further step of generalization, where \mathbf{W} is now a 4D tensor. Intuitively, this formulation allows the label compatibility pattern to be spatially varying across different pixel locations. Eq. 3 can be seen as a special case factorizing the 4D tensor as the product of two 2D matrices.

Mean-field inference derivation We will start from the mean-field update equation for general pairwise CRFs, Eq. 4. Detailed derivation for it can be found in Koller and Friedman [1, Chapter 11.5].

$$Q_i(l) = \frac{1}{Z_i} \exp \left\{ -\psi_u(l) - \sum_{j \in \Omega(i)} \mathbf{E}_{l_j \sim Q_j} \psi_p(l, l_j) \right\} \quad (4)$$

Considering that we use multiple neighborhoods (with different dilation factors) in parallel, the update equation becomes:

$$Q_i(l) = \frac{1}{Z_i} \exp \left\{ -\psi_u(l) - \sum_k \sum_{j \in \Omega^k(i)} \mathbf{E}_{l_j \sim Q_j} \psi_p^k(l, l_j) \right\}. \quad (5)$$

Substituting the pairwise potential with:

$$\psi_p^k(l_i, l_j) = K^k(\mathbf{f}_i, \mathbf{f}_j) \mathbf{W}_{l_j l_i}^k[\mathbf{p}_j - \mathbf{p}_i], \quad (6)$$

the update rule becomes:

$$\begin{aligned} Q_i(l) &= \frac{1}{Z_i} \exp \left\{ -\psi_u(l) \right. \\ &\quad \left. - \sum_k \sum_{j \in \Omega^k(i)} \mathbf{E}_{l_j \sim Q_j} \left\{ K^k(\mathbf{f}_i, \mathbf{f}_j) \mathbf{W}_{l_j l_i}^k[\mathbf{p}_j - \mathbf{p}_i] \right\} \right\} \\ &= \frac{1}{Z_i} \exp \left\{ -\psi_u(l) \right. \\ &\quad \left. - \sum_k \sum_{l' \in \mathcal{L}} \sum_{j \in \Omega^k(i)} K^k(\mathbf{f}_i, \mathbf{f}_j) \mathbf{W}_{l' l_i}^k[\mathbf{p}_j - \mathbf{p}_i] Q_j(l') \right\} \end{aligned} \quad (7)$$

Using Eq. 7 in an iterative fashion leads to the final update rule of mean-field inference:

$$\begin{aligned} Q_i^{(t+1)}(l) &\leftarrow \frac{1}{Z_i} \exp \left\{ -\psi_u(l) \right. \\ &\quad \left. - \underbrace{\sum_k \sum_{l' \in \mathcal{L}} \sum_{j \in \Omega^k(i)} K^k(\mathbf{f}_i, \mathbf{f}_j) \mathbf{W}_{l' l_i}^k[\mathbf{p}_j - \mathbf{p}_i] Q_j^{(t)}(l')}_{\text{PAC}} \right\}. \end{aligned} \quad (8)$$

Mean-field inference steps Tab. 2 shows how mIoU changes with different mean-field steps. We use 5 steps for all other experiments in the paper.

Table 2: **Impact of MF steps in PAC-CRF.** Validation mIoU when using different number of MF steps in PAC-CRF.

Mean-field steps	1	3	5	7
mIoU	68.38	68.72	68.90	68.90
time	19 ms	49 ms	78 ms	109 ms

On the contribution of dilation Just like standard convolution, PAC supports dilation to increase the receptive field without increasing the number of parameters. This capability is leveraged by PAC-CRF to allow long-range connections. For a similar purpose, Conv-CRF applies Gaussian blur to pairwise potentials to increase the receptive field. To quantify the improvements due to dilation, we try another baseline where we add dilation to Conv-CRF. The improved performance (+2.13/+1.57 \rightarrow +2.50/+1.91) validates that dilation is indeed an important ingredient, while the remaining gap shows that the PAC formulation is essential to the full gain.

References

- [1] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. 2
- [2] Y. Li, J.-B. Huang, N. Ahuja, and M.-H. Yang. Deep joint image filtering. In *European Conference on Computer Vision*, pages 154–169. Springer, 2016. 1, 3

