Deep Convolutional Neural Network for Image Deconvolution

Li Xu[1], Jimmy SJ. Ren[1], Ce Liu[2], Jiaya Jia[3]

[1]Lenovo Research & Technology [2]Microsoft Research [3]The Chinese University of Hong Kong

http://lxu.me/projects/dcnn/

Challenges in deconvolution

Saturation Compression Noise

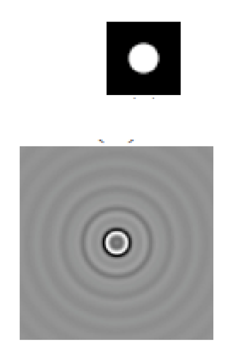
Blur Degradation

Problems of existing methods

Architecture Limitation

->Deconvolution involves much larger Spatial Support

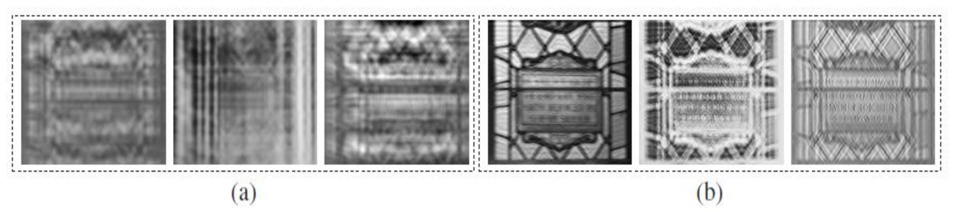
$$x = \mathcal{F}^{-1}\left(\frac{1}{\mathcal{F}(k)}\left\{\frac{|\mathcal{F}(k)|^2}{|\mathcal{F}(k)|^2 + \frac{1}{SNR}}\right\}\right) * y = k^{\dagger} * y,$$



Training Limitation

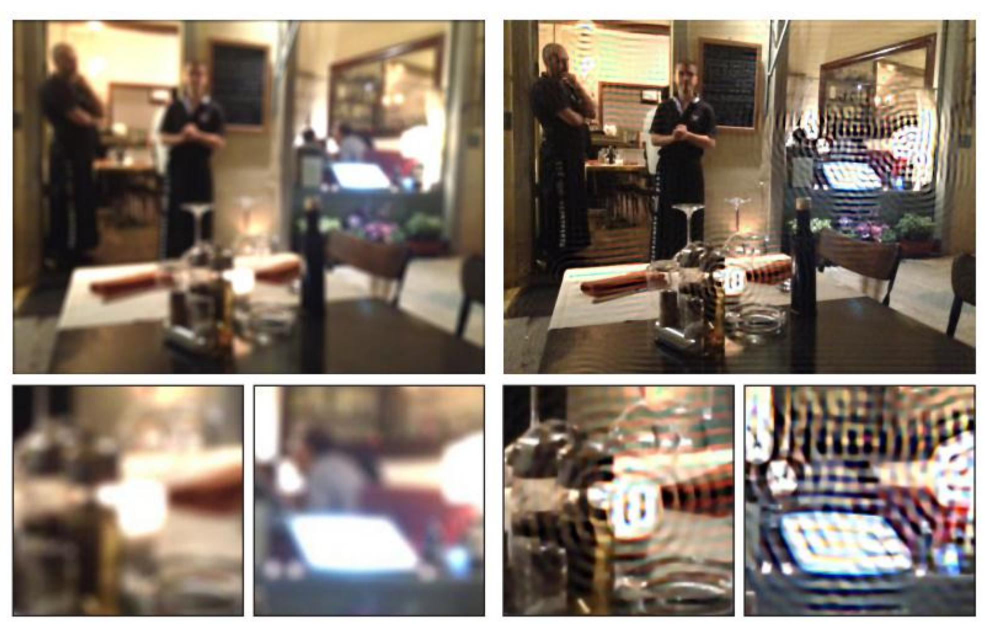
->Difficult to model all the degradations

->Difficult to train a network blindly



(a) (b)

Figure 6: Comparisons of intermediate results from deconvolution CNN. (a) Maps from random initialization. (b) Our final maps.



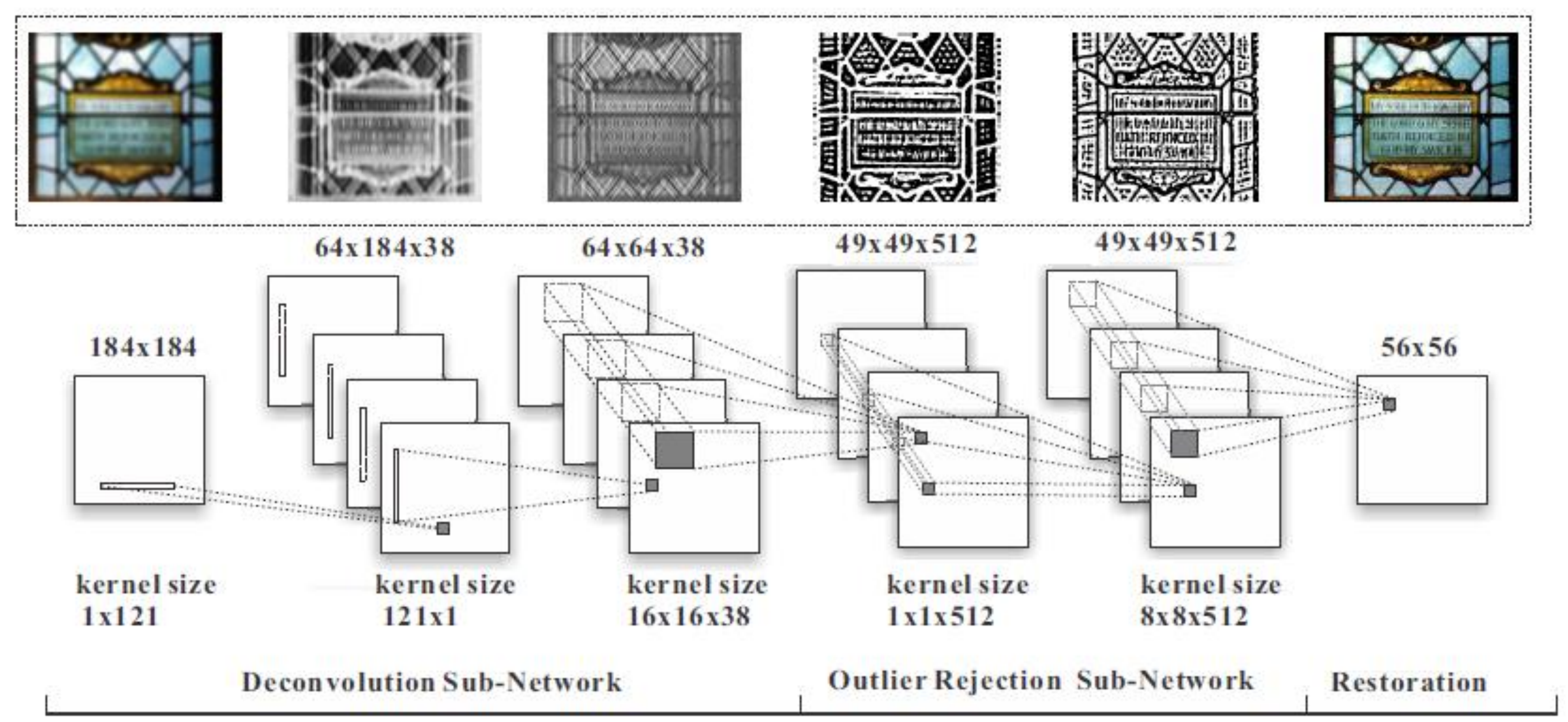
Direct Adoption of Learning Fails

(a) input (b) SSDAE [14] (c) CNN [15] (d) Ours

Overview

Many fundamental image-related problems involve deconvolution operators. Real blur degradation seldom complies with an ideal linear convolution model due to camera noise, saturation, image compression, to name a few. Instead of perfectly modeling outliers, we develop a deep convolutional neural network to capture the characteristics of degradation. Our solution is to establish the connection between traditional optimization-based schemes and a neural network architecture where a novel, separable structure is introduced as a reliable support for robust deconvolution against artifacts. Our novel approach yields decent performance on non-blind image deconvolution compared to previous generative-model based methods.

Our Deconvolution CNN



Two sub-networks, deconvolution CNN (DCNN) and Outlier-rejection deconvolution CNN (ODCNN) trained supervised respectively and merged as a integrated network through fine-tuning. Weights in DCNN is theoretically initialized from the separable kernel inversion.

Theory guided weights initialization

$$x = \mathcal{F}^{-1}\left(\frac{1}{\mathcal{F}(k)}\left\{\frac{|\mathcal{F}(k)|^2}{|\mathcal{F}(k)|^2 + \frac{1}{SNR}}\right\}\right) * y \qquad k^{\dagger} * y = \sum_j s_j \cdot u_j * (v_j^T * y),$$

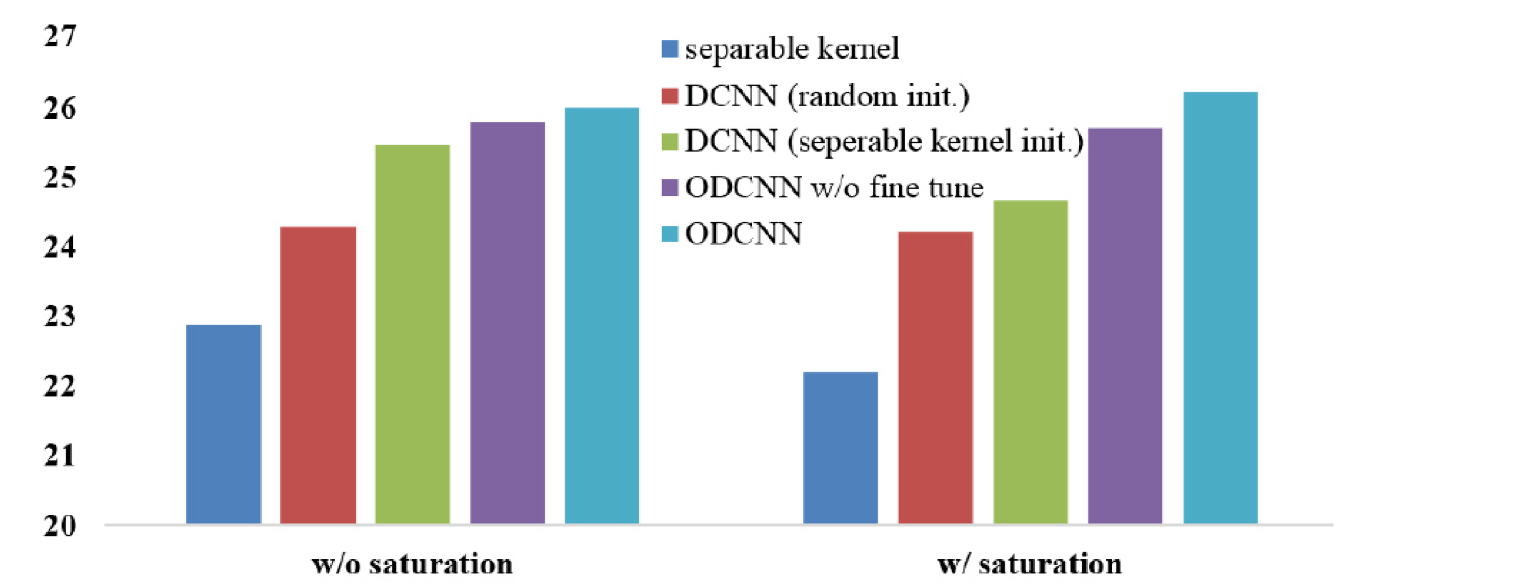
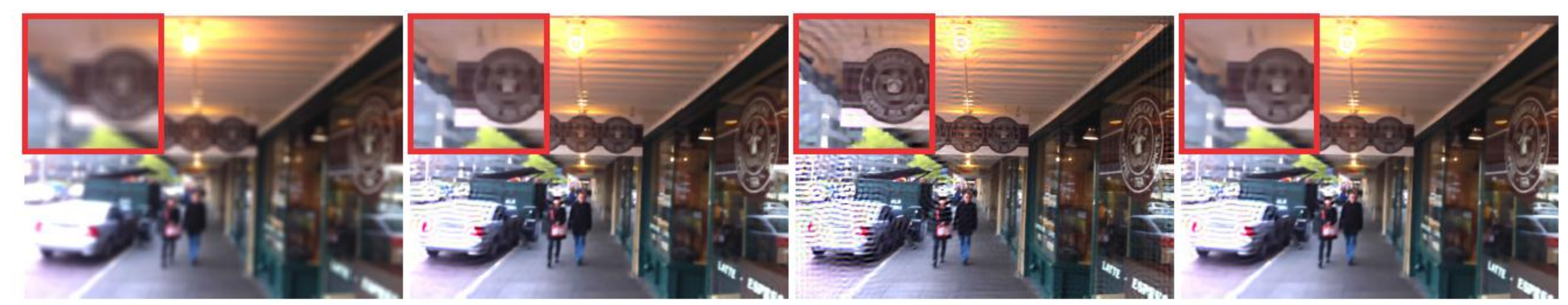


Figure 4: PSNRs produced in different stages of our convolutional neural network architecture.

Our Results



(a) Input (b) Levin et al. [7] (c) Krishnan et al. [3] (d) EPLL [11]

(e) Cho et al. [9] (f) Whyte et al. [10] (g) Schuler et al. [13] (h) Ours

kernel type	Krishnan [3]	Levin [7]	Cho [9]	Whyte [10]	Schuler [13]	Schmidt [4]	Ours
disk sat.	24.05dB	24.44dB	25.35dB	24.47dB	23.14dB	24.01dB	**26.23dB**
disk	25.94dB	24.54dB	23.97dB	22.84dB	24.67dB	24.71dB	**26.01dB**
motion sat.	24.07dB	23.58dB	25.65 dB	25.54dB	24.92dB	25.33dB	**27.76dB**
motion	25.07dB	24.47 dB	24.29dB	23.65dB	25.27dB	25.49dB	**27.92dB**

Table 1: Quantitative comparison on the evaluation image set.