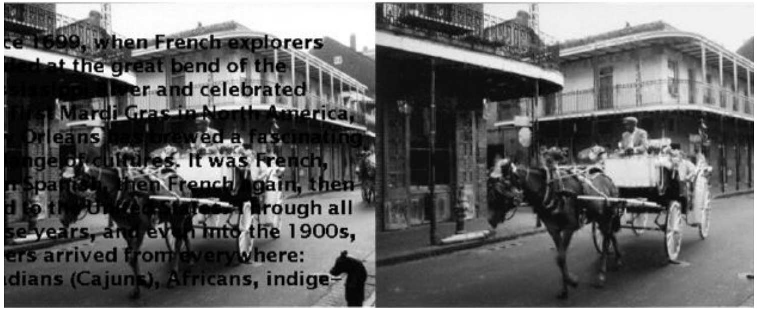
PixelRL: Fully Convolutional Network with Reinforcement Learning for Image Processing

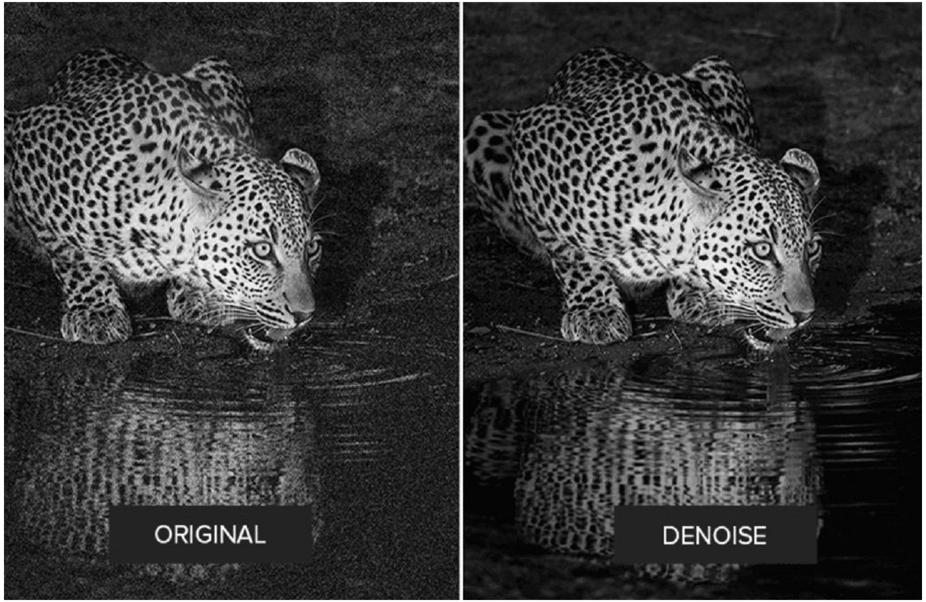
Image Processing

Computer vision focuses on making sense of what the machines see.
Image processing, on the other hand, transform images in many ways such as smoothing, filtering, enhancing, inpainting, blurring, etc..

color enhancing



inpainting



denoising

Fully Convolutional Network with Reinforcement Learning

Denoising

$s^{(0)} = I$ is a input noisy image, the agents iteratively remove the noises by executing actions.

$$r_i^{(t)} = (I_i^{target} - s_i^{(t)})^2 - (I_i^{target} - s_i^{(t+1)})^2,$$

that maximize the mean of the total expected rewards at all pixels:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} E_{\pi} \left(\sum_{t=0}^{\infty} \gamma^t \overline{r}^{(t)} \right), \quad (5)$$

$$\overline{r}^{(t)} = \frac{1}{N} \sum_{i=1}^{N} r_i^{(t)}, \quad (6)$$

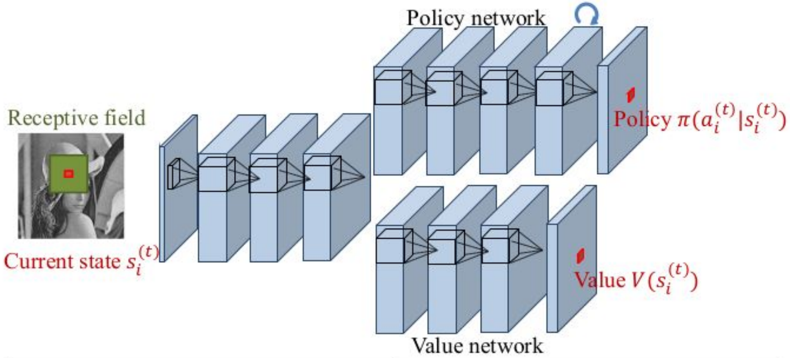
where $\overline{r}^{(t)}$ is the mean of the rewards $r_i^{(t)}$ at all pixels.

TABLE I: Actions for image denoising and restoration.

	action	filter size	parameter
1	box filter	5x5	-
2	bilateral filter	5x5	$\sigma_c = 1.0, \sigma_S = 5.0$
3	bilateral filter	5x5	$\sigma_c = 0.1, \sigma_S = 5.0$
4	median filter	5x5	-
5	Gaussian filter	5x5	$\sigma = 1.5$
6	Gaussian filter	5x5	$\sigma = 0.5$
7	pixel value += 1	-	-
8	pixel value -= 1	-	-
9	do nothing	-	-

Fully Convolutional Network with Reinforcement Learning

Architecture



Shared network				Policy network			
Conv +ReLU	Conv +ReLU	Conv +ReLU	Conv +ReLU	Conv +ReLU	Conv +ReLU	ConvGRU	Conv +Softmax
3x3, 1, 64	3x3, 2, 64	3x3, 3, 64	3x3, 4, 64	3x3, 3, 64	3x3, 2, 64	3x3, 1, 64	3x3, 1, $\|A\|$
				Value network			
				Conv +ReLU	Conv +ReLU	Conv	
				3x3, 3, 64	3x3, 2, 64	3x3, 1, 1	

Fig. 1: Network architecture of the fully convolutional A3C. The numbers in the table denote the filter size, dilation factor, and output channels, respectively.

Objective

$$r_i^{(t)} = (I_i^{target} - s_i^{(t)})^2 - (I_i^{target} - s_i^{(t+1)})^2,$$

that maximize the mean of the total expected rewards at all pixels:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, E_{\pi}\left(\sum_{t=0}^{\infty} \gamma^t \overline{r}^{(t)}\right), \tag{5}$$

$$\overline{r}^{(t)} = \frac{1}{N}\sum_{i=1}^{N} r_i^{(t)}, \tag{6}$$

where $\overline{r}^{(t)}$ is the mean of the rewards $r_i^{(t)}$ at all pixels.

A3C

$$R_i^{(t)} = r_i^{(t)} + \gamma V(s_i^{(t+1)}),$$

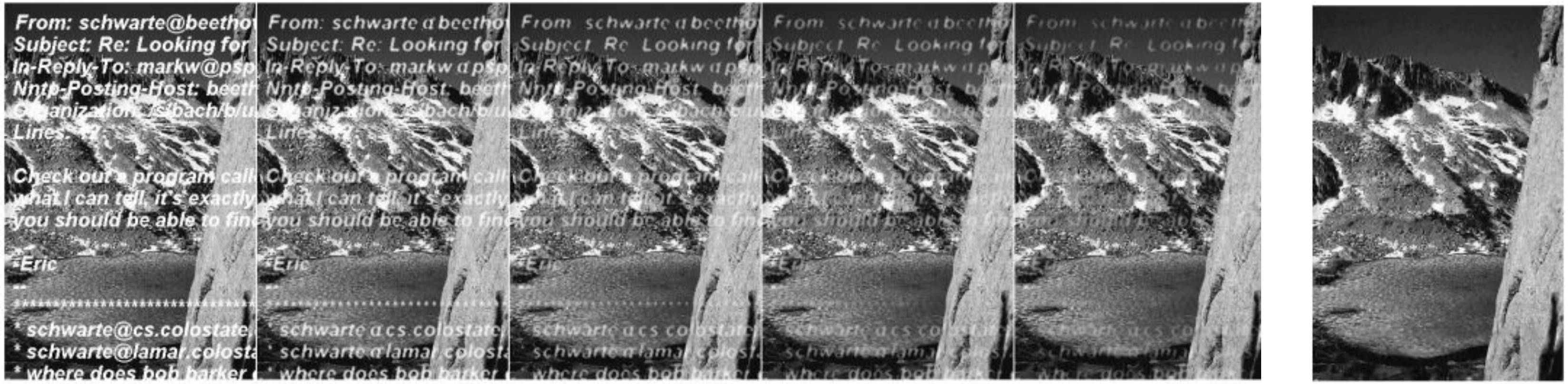
$$d\theta_v = \nabla_{\theta_v} \frac{1}{N}\sum_{i=1}^{N}\left(R_i^{(t)} - V(s_i^{(t)})\right)^2,$$

Improving: Reward Map Convolution

$$R_i^{(t)} = r_i^{(t)} + \gamma \sum_{j \in \mathcal{N}(i)} w_{i-j} V(s_j^{(t+1)}),$$

Fully Convolutional Network with Reinforcement Learning

Inpainting



$t = 0$ (Input) $t = 1$ $t = 2$ $t = 3$ $t = 4$ target

Color Enhancing

$t = 0$ (Input) $t = 1$ $t = 2$ $t = 3$ target

Datasets

BSD68. contains **493** images (428 train and 65 test)

dl-image-enhance. contains **112** images (67 train and 45 test)

input **edited**

Evaluation

Color Enhancing

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2.$$

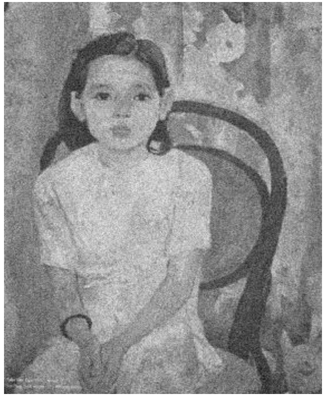
The PSNR (in dB) is defined as

Denoising and Inpainting

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right)$$

Demo & Fail cases

Denoising

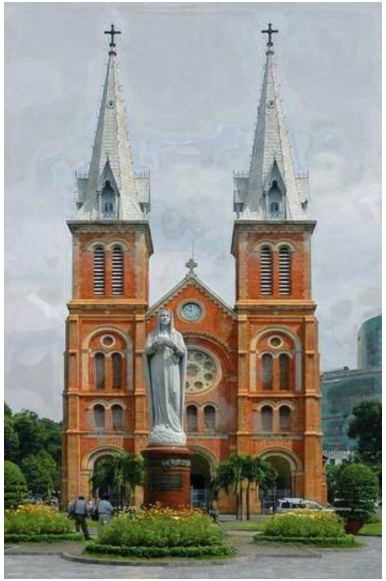



Demo & Fail cases

Inpainting

Demo & Fail cases

Color Enhancing



References

PixelRL: Fully Convolutional Network with Reinforcement Learning for Image Processing