CS331.M21.KHCL

Final Report: Fully Convolutional Network with Reinforcement Learning for Image Processing

Cat Van Tai 19522147
Do Hoang Phuc 19522028

June 2022

PhD Nguyen Vinh Tiep

Contents

1 Introduction

1.1 Image Processing

Image Processing deals with manipulation of digital images through a digital computer. It is a subfield of signals and systems but focus particularly on images. It focuses on developing a computer system that is able to perform processing on an image. The input of that system is a digital image and the system process that image using efficient algorithms, and gives an image as an output. The tasks of image processing are such as compressing, smoothing, filtering, enhancing, inpainting, blurring, etc.. In this report, we will focus on three tasks are denoising, inpainting and color enhancing. Our approach that using Reinforcement Learning to solve them.

Image Denoising where the underlying goal is to estimate the original image by suppressing noise from a noise-contaminated version of the image. Image noise may be caused by different intrinsic (i.e., sensor) and extrinsic (i.e., environment) conditions which are often not possible to avoid in practical situations. Therefore, image denoising plays an important role in a wide range of applications such as image restoration, visual tracking, image registration, image segmentation, and image classification, where obtaining the original image content is crucial for strong performance.

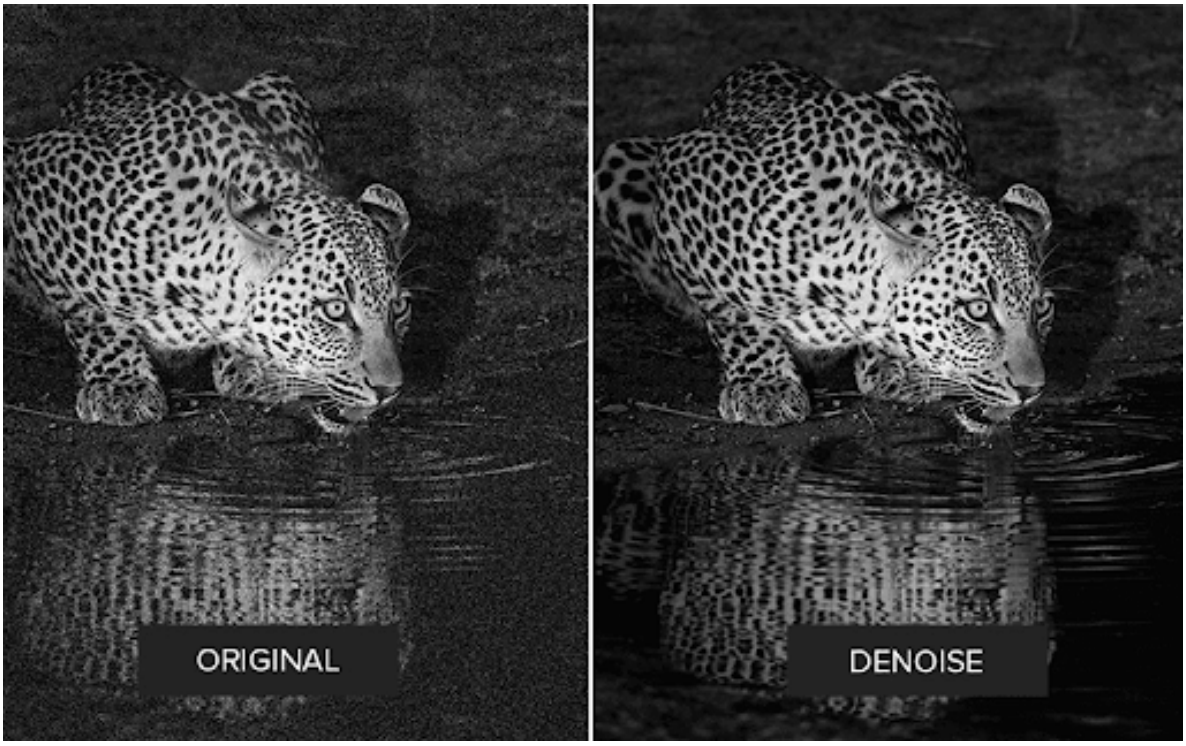


Figure 1: *Image Denoising*

Image Inpainting the goal of this task find out the semantic content that should be filled for missing part. This research field has achieved promising progress by using neural image inpainting methods. Nevertheless, there is still a critical challenge in guessing the missed content with only the context pixels.

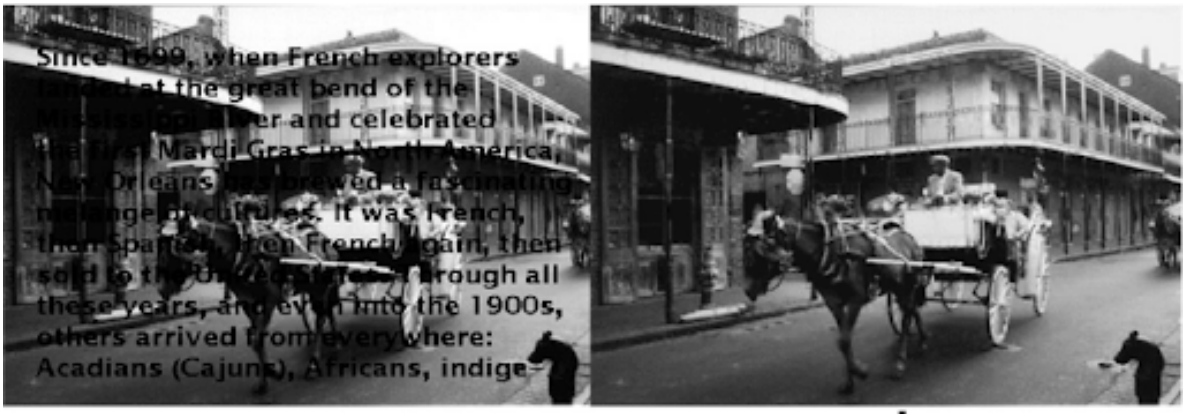


Figure 2: *Image Inpainting*

Color Enhancement are process consists of a collection of techniques that seek to improve the visual appearance of an image or to convert the image to a form a better suited for analysis by a human or machine.

Figure 3: *Color Enhancing*

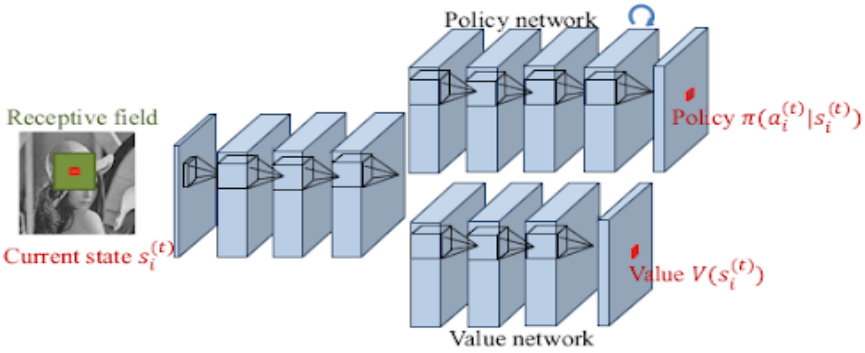
1.2 Restriction and Motivation

In recent years, deep learning has been achieving great success not only in image classification, but also in image processing tasks such as image filtering, colorization, generation, and translation. Although all of the above employ neural networks to learn the relationships between the input and output, their structures are approximately divided into two categories. The neural networks typically

used for image recognition are convolutional neural networks (CNNs) that have a number of convolution layers followed by fully-connected layers to output the classification scores. In contrast, fully convolutional networks (FCNs) that do not have fully connected layers are employed for image processing tasks because a pixel-wise output is required for their tasks.

After the introduction of the deep Q-network (DQN), which can play Atari games on the human level, a lot more attention has been focused on deep reinforcement learning (RL). Recently, deep RL has also been applied to a variety of image processing tasks. However, these methods can execute only global actions for the entire image and are limited to simple applications. To overcome this drawback, they propose a new problem setting: pixelRL for image processing. PixelRL is a multi- agent RL problem, where each pixel has an agent. The agents learn the optimal behavior to maximize the mean of the expected total rewards at all pixels. Each pixel value is regarded as the current state and is iteratively updated by the agent's action.

2 Methodology



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

Figure 4: *Network architecture of the fully convolutional A3C. The numbers in the table denote the filter size, dilation factor, and output channels, respectively.*

2.1 Pixel-wise rewards

Let I_i be the i-th pixel in the input image I that has N pixels ($i = 1, .., N$). Each pixel has an agent, and its policy is denoted as $\pi_i(a_i|s_i)$, where $a_i^{(t)} (\in A)$ and $s_i^{(t)}$ are the action and the state of the i-th agent at the time step t, respectively. A is the predefined action set, and $s_i = I_i$. The agents obtain the next states $s^{(t+1)} = (s_1^{(t+1)}, .., s_N^{(t+1)})$ and rewards $r^{(t)} = (r_1^{(t)}, .., r_N^{(t)})$ from the environment by taking the actions $a^{(t)} = (a_1^{(t)}, .., a_N^{(t)})$. The objective of the pixelRL problem is to learn the optimal

policies $\pi = (\pi_1, .., \pi_N)$ that maximize the mean of the total expected rewards at all pixels:

$$\pi^* = argmax_\pi E_\pi(\sum_{t=0}^{\infty} \gamma^t \bar{r}^{(t)})$$

, where $\bar{r}^{(t)}$ is the mean of the rewards r_i at all pixels:

$$\bar{r}^{(t)} = \frac{1}{N} \sum_{i=1}^{N} r_i^{(t)}$$

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

, where I_i^{target} is the i-th pixel value of the original clean image.

2.2 Fully convolution network

Training N agents is also computationally impractical when the number of pixels is large. And it treats only the fixed size of images. To solve the problems, they employ a FCN instead of N networks. By using the FCN, all the N agents can share the parameters, and they can parallelize the computation of N agents on a GPU, which renders the training efficient. In addition, they employ A3C and extend it to the fully convolutional form.

2.3 Reward map convolution

When the receptive fields of the FCNs are 1x1 (i.e., all the convolution filters in the policy and value network are 1x1),the N subproblems are completely independent. In this case, they proposed reward map convolution, the policy and value networks observe not only the i-th pixel $s_i^{(t)}$ but also the neighbor pixels to output the policy π and value V at the i-th pixel. In other words, the action $a_i^{(t)}$ affects not only the state $s_i^{(t)}$ but also the policies and values in $N(i)$ at the next time step, where $N(i)$ is the local window centered at the i-th pixel. Therefore, to consider it, we replace $R_i^{(t)}$ as follows:

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

where w_{ij} is the weight that means how much we consider the values V of the neighbor pixels at the next time step $(t+1)$. w can be regarded as a convolution filter weight and can be learned simultaneously with the network parameters θ_p and θ_v.

2.4 Actions for Color Enhancing

action		
1 / 2	contrast	x0.95 / x1.05
3 / 4	color saturation	x0.95 / x1.05
5 / 6	brightness	x0.95 / x1.05
7 / 8	red and green	x0.95 / x1.05
9 / 10	green and blue	x0.95 / x1.05
11 / 12	red and blue	x0.95 / x1.05
13	do nothing	-

2.5 Actions for image denoising and restoration

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

3 Experiment

3.1 Dataset

BSD68 dataset. For image denoising and image inpainting task, we use BSD68 dataset for image denoising benchmarks is part of The Berkeley Segmentation Dataset and Benchmark. It is used for measuring image denoising algorithms performance. It contains 428 train images and 68 test images.

dl-image-enhance. We used the dataset , which has 70 train images and 45 test images downloaded from Flicker. Using Photoshop, all the images were enhanced by a professional photographer for three different stylistic local effects: Foreground Pop-Out, Local Xpro, and Watercolor.

3.2 Evaluation

For image denoising and image inpainting task, we use PSNR metric to evaluate model. In the other hand, we use MSE metric for evaluating color enhancing task.

$$PSNR = 10.\log_{10}(\frac{MAX_I^2}{MSE})$$

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

3.3 Results

3.3.1 Image Denoising

Table 1: PSNR [dB] on BSD68 test set with Gaussian noise for image denoising task

Method	15	25	50
CNN	**31.66**	**29.18**	**26.20**
PixelRL	31.49	28.94	25.95

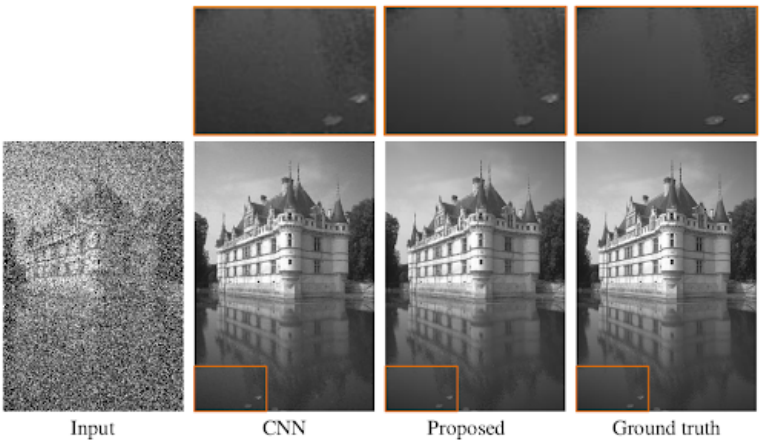


Figure 5: *Qualitative comparison of CNN, the proposed method and Ground truth*

3.3.2 Image Inpainting

Table 2: Comparison on image restoration

Method	PSNR [dB]
CNN	29.75
PixelRL	**29.97**

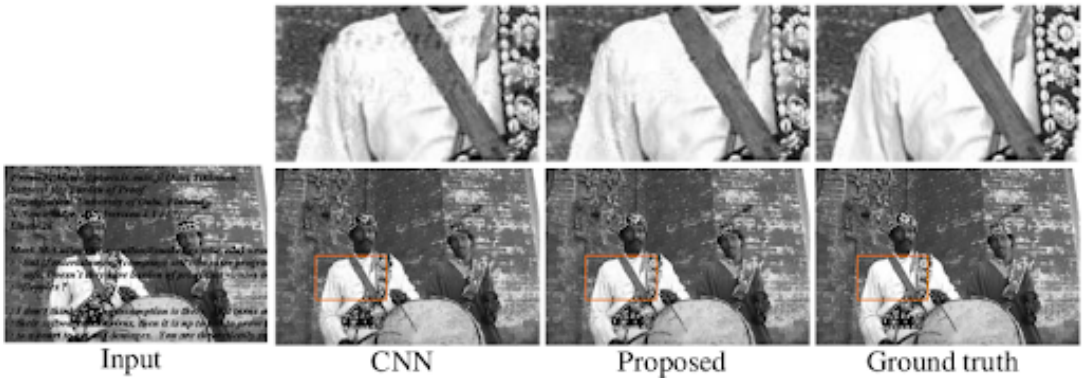


Figure 6: *Qualitative comparison of CNN, the proposed method and Ground truth*

3.3.3 Color Enhancing

Table 3: Comparison of mean L2 testing errors on local color enhancement. The errors of Pix2Pix and PixelRL

Method	Foreground Pop-Out
DNN	7.43
PixelRL	**5.67**

Figure 7: *Qualitative comparison of DNN, the proposed method and Ground truth*

4 Conclusions

The experimental results demonstrated that the proposed method achieved comparable or better results than state-of-the-art methods on each application.

Different from the existing deep learning-based methods for such applica- tions, the proposed method is interpretable. The interpretability of deep learning has been attracting much attention and it is especially important for some applications such as medical image processing.

5 Reference

[1] R Furuta, N Inoue, T Yamasaki. PixelRL: Fully Convolutional Network with Reinforcement Learning for Image Processing.