Fast Image Processing

2018 Digital Image Processing Term Project

I. Introduction

Image processing can be used in everywhere. Like robotic navigation systems, traffic, healthcare, camera edge technology, etc. Faster image processing can save us tons of time. Learning how to accelerate the procedure of image processing is undoubtedly worthy.
In our poster, we present an approach by using fully convolutional network to accelerate some image processing operators and evaluate the performance of our trained model, including pencil drawing, multi-scale toning and nonlocal dehazing, etc.

II. Methodology

Problem Definition

Given
$\mathbf{I}$: image in RGB color space
f: operator

Goal
- To approximate f with another operator $\hat{f}$, that is

$$\hat{f}(\mathbf{I}) \approx f(\mathbf{I})$$

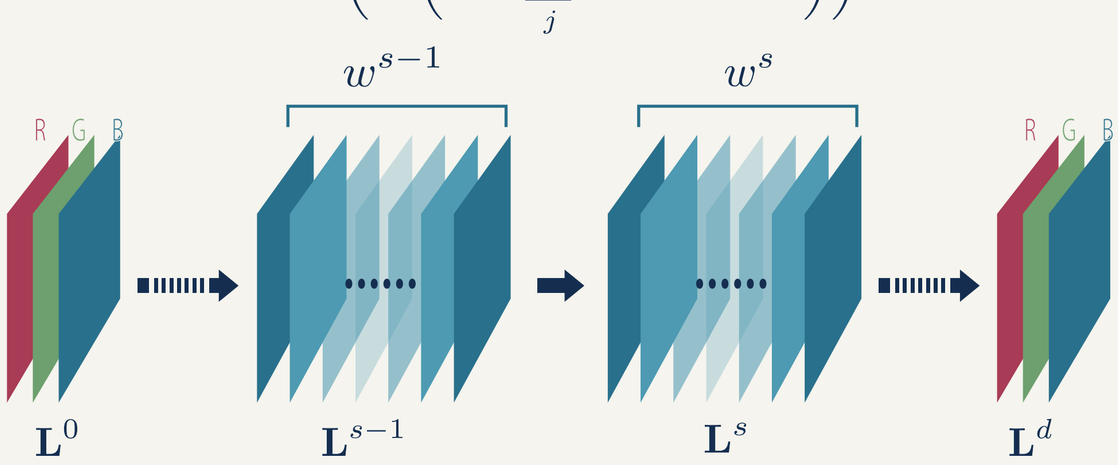
- To find a broadly applicable image processing operator

Context Aggregation Networks

The primary architecture we used is multi-scale context aggregation network (CAN).
Now let's describe the parameterization in detail!

The image data lay out over layers: $\{\mathbf{L}^0, \ldots, \mathbf{L}^d\}$
- Dimensions of $\mathbf{L}^0$ (input) and $\mathbf{L}^d$ (output): $m \times n \times 3$
- Dimensions of $\mathbf{L}^s$ $(1 \leq s \leq d-1)$: $m \times n \times w$
  $\mathbf{L}^s$ is computed from the previous layer $\mathbf{L}^{s-1}$:

$$\mathbf{L}_i^s = \Phi\left(\Psi^s\left(b_i^s + \sum_j \mathbf{L}_j^{s-1} *_{r_s} \mathbf{K}_{i,j}^s\right)\right)$$



and $\Psi^s(x)$ is computed as follow:

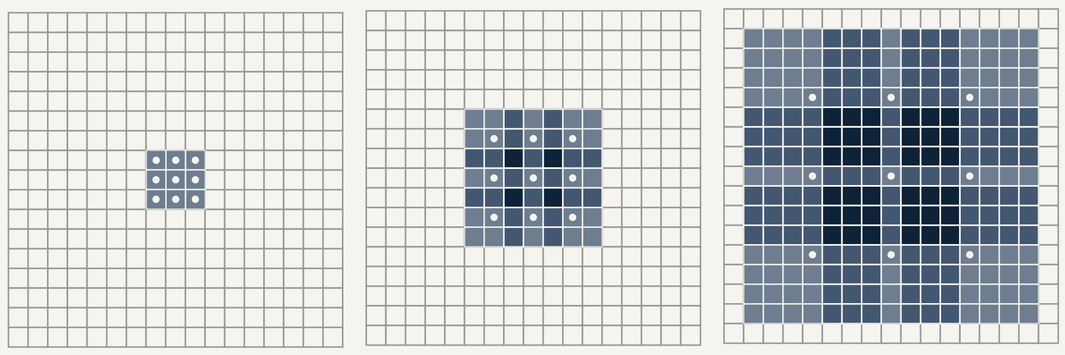
$$\Psi^s(x) = \lambda_s x + \mu_s BN(x)$$

where $\lambda_s, \mu_s \in \mathbb{R}$ are learned scalar weights and BN is the batch normalization operator. We'll keep training these two scalars for each iteration.

Specifically, for image coordinates $\mathbf{x}$:

$$(\mathbf{L}_j^{s-1} *_{r_s} \mathbf{K}_{i,j}^s)(\mathbf{x}) = \sum_{\mathbf{a}+r_s\mathbf{b}=\mathbf{x}} \mathbf{L}_j^{s-1}(\mathbf{a})\mathbf{K}_{i,j}^s(\mathbf{b})$$

Performing dilated convolution needs to specify the kernel size, it looks like:



1-dilated | 2-dilated | 3-dilated

III. Implementation

Training

The network is trained on a set of input-output pairs that contain images before and after the application of original operator: $\mathcal{D} = \{\mathbf{I}_i, f(\mathbf{I}_i)\}$.

The parameters of the network are the kernel weights $\mathcal{K} = \{\mathbf{K}_{i,j}^s\}_{s,i,j}$ and the scalar biases $\mathcal{B} = \{b_i^s\}_{s,i}$.

These parameters are optimized to fit the action of the operator f across all images in the training set. We train with an image-space regression loss:

$$\ell(\mathcal{K}, \mathcal{B}) = \sum_i \frac{1}{N_i} ||\hat{f}(\mathbf{I}_i; \mathcal{K}, \mathcal{B}) - f(\mathbf{I}_i)||^2,$$

where N_i is the number of pixels in image $\mathbf{I}_i$.

IV. Results

Run time with 2,500 images (256 x 256 x 3)

Operator	Our Result w/o GPU	Our Result w/ GPU (NVIDIA GeForce GTX 1080)
2,919 (sec)	1,814 (sec)	70 (sec)

Pencil Drawing

Original | Our Result | Operator

Multi-scale Toning

Original | Our Result | Operator

Nonlocal Dehazing

Hazed | Original | Our Result | Operator

V. Conclusion & Future Work

We implement the CAN based deep learning model to dramatically reduce the image processing time and still get good results and even do better than original operators.
We hope we can find better parameters and neural network to enhance the image quality since in a few cases we still lose qualities even the time cost is low.

VI. Reference

[1] https://arxiv.org/abs/1709.00643 - Fast Image Processing with Fully-Convolutional Networks
[2] https://arxiv.org/pdf/1511.07122.pdf - Multi-scale context aggregation by dilated convolutions
[3] https://github.com/CQFIO/FastImageProcessing

Yu-Sheng Lin
R06945003

Peng-Yu Chen
B03902129

National Taiwan University