Digital Image Processing, Spring 2018

Proposal Report
DUE DATE: May 15, 2018

Team 10: R06945003 林鈺盛 B03902129 陳鵬宇

Paper title

Title: Fast Image Processing with Fully-Convolutional Networks
Conference: ICCV 2017
Authors: Qifeng Chen, Jai Xu Intel Labs and Vladlen Koltun

Motivation

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.

Problem definition

Use convolution neural network (CNN) to implement image processing methods mentioned in class.

Algorithm

Preliminaries

Given:

- $\mathbf{I}$: image in RGB space and
- f: operator, where $\mathbf{I}$ and $f(\mathbf{I})$ have the same resolution.

Our goal is:

- to approximate f with another operator $\hat{f}$, that is $f(\mathbf{I}) \approx \hat{f}(\mathbf{I})$ for all images and
- to find a broadly applicable image processing operator.

There are three desirable criteria we want to meet with:

1. Accuracy
2. Speed
3. Compactness

Context Aggregation Networks (CAN)

The primary architecture the paper used is **multi-scale context aggregation network (CAN)**
Now let's describe the parameterization in detail.
The data lay out over layers: $\{\mathbf{L}^0, \ldots, \mathbf{L}^d\}$.

- Dimension of $\mathbf{L}^0$ (input) and $\mathbf{L}^d$ (output): $m \times n \times 3$.
- Dimension of $\mathbf{L}^s$ $(1 \leq s \leq d-1)$: $m \times n \times w$, where w is the width (features) of each layer.

$\mathbf{L}^s$ can be computed from the previous layer $\mathbf{L}^{s-1}$ as follows:

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

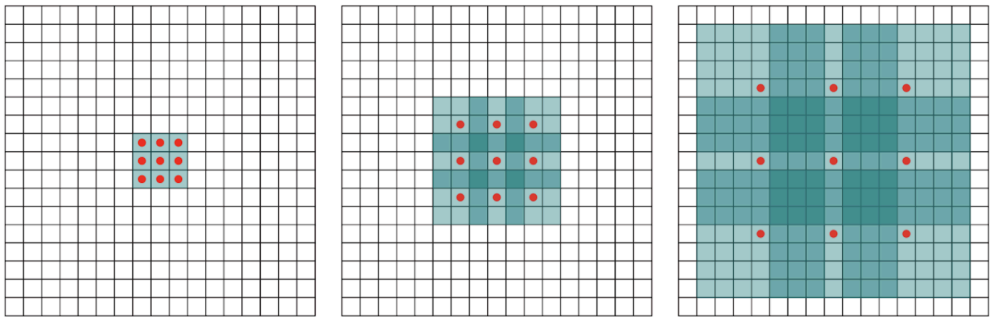
and $\Psi^s(x)$ is computed as follow:

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

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.
Specically, for image coordinates x:

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

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



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 $\mathcal{B} = \{b_i^s\}_{s,i}$.

Therse 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 $\mathbf{I}_i$.

Extensions

Parameterized operators

Image processing operator can have parameters that control its action.

- image smoothing operator: λ
- higher λ leads to more aggressive smoothing

Therefore, we can add an input channel that is used to communicate the parameter's value to the network.

Single network

- Augment the input layer by adding 10 additional channels, where each channel is a binary indicator that corresponds to one of the 10 operators.
- Remarkably, a single compact network that represents all 10 operators achieves high accuracy.

Expected results

Use CNN model to implement ***edge detection, half toning, noise removal***, etc.
Here is the result of the paper:

Reference

[1] Fast Image Processing with Fully-Convolutional Networks
[2] Multi-scale context aggregation by dilated convolutions