Assignment #2

B05902120 / Yu-Ting, TSENG

Nov 3, 2019

Semantic Segmentation

For the baseline model, I normalize the data with the mean value and standard value of ImageNet dataset and random flip the image with probability of 0.5 as data augmentation. Worth noting is that I only flip the image of `training` set, rather than both `training` and `validation` set.

```
transform = transforms.Compose([
            transforms.RandomHorizontalFlip(0.5),
            transforms.ToTensor(),
            transforms.Normalize(MEAN, STD)])
```

The value of training loss and mIOU score over various epoch is showing in the figure as below. The training loss converges to approximately 0.25; on the other hand, the mIOU score coverges to about 0.55.

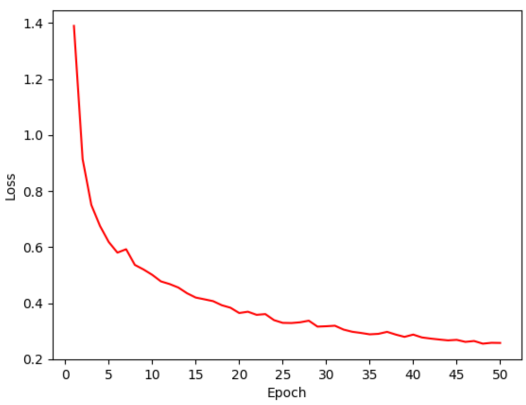

Figure 1: Original

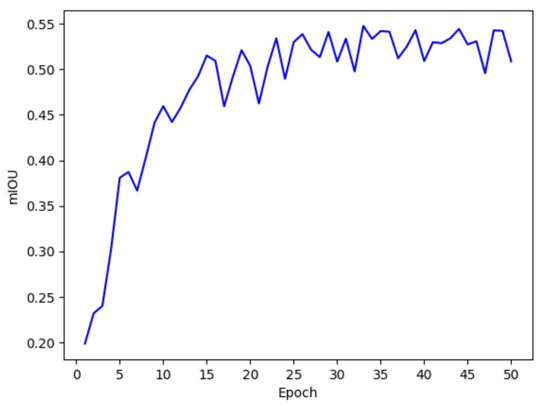

Figure 2: Result

For the improved model, I do the same steps for preprocessing the data, random flip and normalize. Next, I change `ReLU` to `LeakyReLU` which can avoid the situation of dead ReLU. In addition, I normalize the data and drop redundant part in every level.

```
self.conv = nn.Sequential(
            nn.ConvTranspose2d(512, 256, kernel_size=4,
               stride=2, padding=1, bias=False),
            nn.LeakyReLU(),
            nn.BatchNorm2d(256),
            nn.Dropout(0.2))
```

Image Filtering

In the lectures, we introduced the concept of image filtering and its applications. In this problem, you are asked to implement basic Gaussian filters and apply them to images for evaluation. Below are the 1D and 2D kernels of a Gaussian filter:

1D kernel: $G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$

2D kernel: $G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$

Given a variance σ^2, the convolution of a 2D Gaussian kernel can be reduced to two sequential convolutions of a 1D Gaussian kernel. Show that convolving with a 2D Gaussian filter is equivalent to sequentially convolving with a 1D Gaussian filter in both vertical and horizontal directions.

$$
\begin{aligned}
G(x, y) &= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} * \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} \\
&= G(x) * G(y)
\end{aligned}
$$

Implement a discrete 2D Gaussian filter using a $3 * 3$ kernel with $\sigma \approx \frac{1}{2\ln 2}$. Use the provided `lena.png` as input, and plot the output image in your report. Briefly describe the effect of the filter.

We can obviously find that gaussian filter reduces the noise of the image, however, it reduces the details on the contrary.

Consider the image $I(x, y)$ as a function $I : \mathbb{R}^2 \leftarrow \mathbb{R}$. When detecting edges in an image, it is often important to extract information from the derivatives of pixel values. Denote

Figure 3: Original

Figure 4: Result

the derivatives as follows:

$$I_x(x,y) = \frac{\partial I}{\partial x} \approx \frac{1}{2}(I(x+1,y) - I(x-1,y))$$
$$I_y(x,y) = \frac{\partial I}{\partial y} \approx \frac{1}{2}(I(x,y+1) - I(x,y-1))$$

Implement the 1D convolution kernels $k_x \in \mathbb{R}^{1*3}$ and $k_y \in \mathbb{R}^{3*1}$ such that

$$I_x = I * k_x$$
$$I_y = I * k_y$$

Write down your answers of k_x and k_y. Also, plot the resulting images I_x and I_y using the provided `lena.png` as input.

$$k_x = [-1, 0, 1]^T$$
$$k_y = [-1, 0, 1]$$

We might be able to find that k_x is good at find the horizontal edge; on the other hand, k_y is excellent for the vertical one.

Define the gradient magnitude image I_m as

$$I_m(x,y) = \sqrt{I_x(x,y)^2 + I_y(x,y)^2}$$

Use both the provided `lena.png` and the Gaussian-filtered image you obtained in 2. as input images. Plot the two output gradient magnitude images in your report. Briefly

Figure 5: I_x

Figure 6: I_y

explain the differences in the results.

Due to the gaussian filter, the detail of the image has been reduced, and thus, the gradient value of two sides of the edge might become smaller than the original one.

Figure 7: Gradient of Ori Image

Figure 8: Gradient of New Image