Weekly Work Report

Yufeng Jiang

VISION@OUC

August 26, 2018

1 Research problem

The main work is understanding the paper [3] and running the code [1] as the basic result compared to the new result with modified codes written by myself.

2 Research approach

Read some papers including GAN, pix2pix, dilation convolution and self-attention *et al.* Write or modify codes cloned from github into the basic code to get more high-resolution and photo-reality images.

3 Research progress

I have learned to write the autoencoder code and GAN code. It is important to read the pix2pix code downloaded from [2], and understand the core network architecture and training process as the basic of understanding [1]. After running the code [1], I have saved the results at checkpoints folder. And then, in order to improve the generated image quality, I have added a novel generator network with dilation convolution.

4 Progress in this week

In this week, I have run the basic code and modified code at the server to get the training results and *.pkl* folders, and then, testing it separately.

4.1 Generator with dilated convolution layers

Drawing inspiration from [5], I decided to add a dilation convolution layers in the generator because dilation convolution is a powerful tool that can enlarge the receptive field of feature points without reducing the resolution of the feature maps. This makes it possible to produce a more high-resolution and photo-reality image.

The whole network in this paper is shown at Figure 1. This kind of complex but shallow network is easy and stable to train. The ResNet34 is originally designed for classification task on mid-resolution images of size 2256×256, but in [5], it is used at the images of size 1024×1024. In our code, we also use it to address images of size 256×256. The center part use dilated convolution layers with skip connection.

If the dilation rates of the stacked dilated convolution layers are 1, 2, 4, 8 respectively, then the receptive field of each layer will be 3, 7, 15, 31. So, the feature points on the last center part layer will see 31×31 points on the first center feature map, covering main part of the main part of the first center feature map.

The decoder of D-LinkNet uses transposed convolution layers to do upsampling, restoring the resolution of feature map from 8×8 to 256×256.

4.2 Results

I trained the code with D-LinkNet network as our generator at the facades datasets. Generated images look a little more clear and real compared to the original resuls. There is a problem knowed just now that the result lacks some *.pkl* folders to as a model for testing. Thus, adding some codes like the following one to get *.pkl* folders is necessary.

```
# save the generator
if epoch % 40 == 0:
    torch.save(self.G.state_dict(), '%s/generator_epoch_%d.pkl' % (self.config['outf'],
                                                                   epoch))
```

Test results using epoch 160 model with the sentence:
python test.py −−config configs/facades.yaml −−modeldir checkpoints/generator_epoch_*160.pkl −*

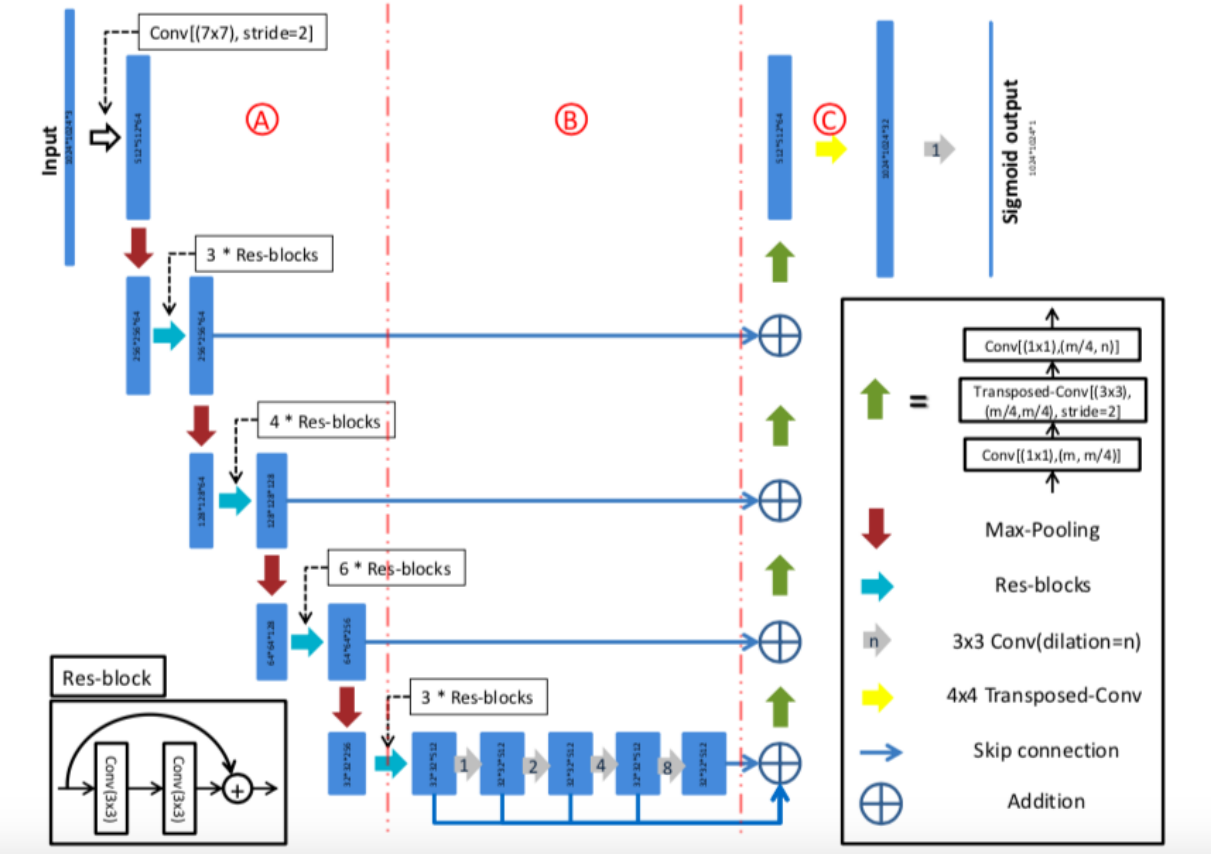

Figure 1: D-LinkNet architecture. Each blue rectangular block represents a multi-channel features map. Part A is the encoder of D-LinkNet. D-LinkNet uses ResNet34 as encoder. Part C is the decoder of D-LinkNet, it is set the same as LinkNet decoder. Original LinkNet only has Part A and Part C. D-LinkNet has an additional Part B which can enlarge the receptive field and as well as preserve the detailed spatial information. Each convolution layer is followed by a ReLU activation except the last convolution layer which use sigmoid activation.

Figure 2: **Left:** The real image. **Center:** The generated fake image using dilated convolution network. **Right:** The generated fake image using basic model.

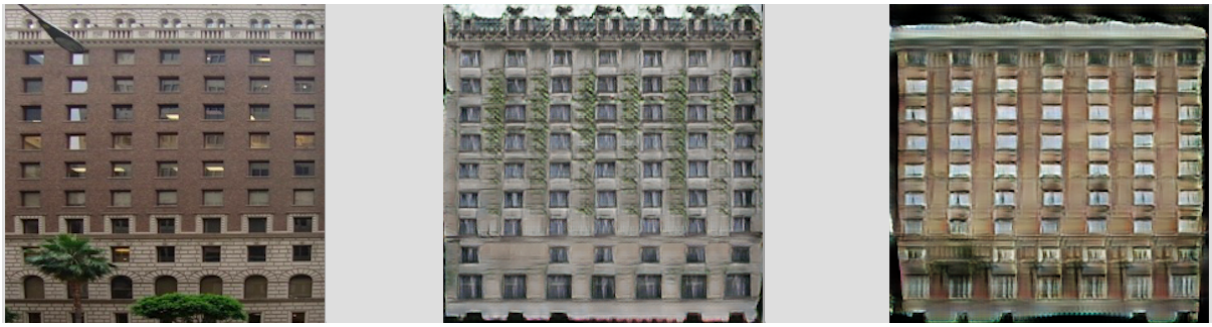

Figure 3: **Left:** The real image. **Center:** The generated fake image using dilated convolution network. **Right:** The generated fake image using basic model.

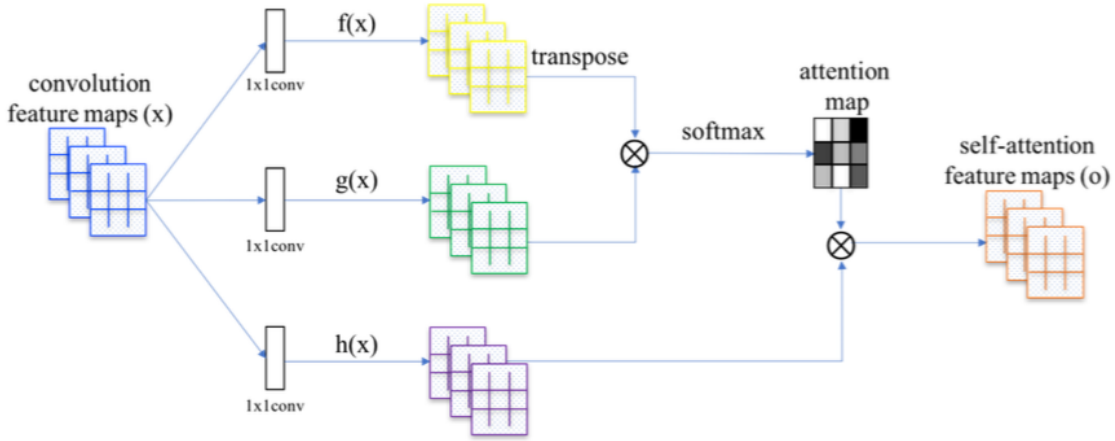

Figure 4: The proposed self-attention mechanism. The $\otimes$ denotes matrix multiplication. The softmax operation is performed on each row.

$-cuda \ --gpu\_ids \ 0$

The realB image, fakeB image generated from dialted convolution network and fakeB image generated by basic code are shown at Figure 2 and Figure 3. From these two images, the generator using dilated convolution layers looks a little bit better than the basic model in my view. However, this is running at facades datasets with short images, so the result may be not accurate exactly. Now, I am running the cityscape with much more images to get a more accurate result.

4.3 Self-attention

In [4], authors propose the self-attention generative adversarial network (SAGAN) which allows attention-driven, long-range dependency modeling for image generation tasks. In SAGAN, details can be generated using cues from all feature locations. Moreover, the discriminator can check that highly detailed features in distant portions of the image are consistent with each other. The network structure is shown at Figure 4. The main thought of this network is adapting the non-local model to add self-attention to the GAN framework, enabling both the generator and the discriminator to efficiently model relationships between widely separated spatial regions. But, I have no idea on how to use it properly.

References

[1] GitHub_godisboy. https://github.com/godisboy/DRPAN. 1

[2] GitHub_junyanz. https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix. 1

[3] C. Wang, H. Zheng, Z. Yu, Z. Zheng, Z. Gu, and B. Zheng. Discriminative region proposal adversarial networks for high-quality image-to-image translation. In *ECCV*, 2018. 1

[4] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv:1805.08318*, 2018. 3

[5] L. Zhou, C. Zhang, and M. Wu. D-LinkNet: LinkNet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In *CVPR*, 2018. 1