DLCV hw4 report
B09901062 黃宥翔

# Problem 1: 3D Novel View Synthesis (50%)

**(5%) Please explain:**

**a. the NeRF idea in your own words**

The NeRF is to train a dense network that transform a 5D positional and angular data to RGB, density data. And with the RGB, density data at each point, we can render the image through volume rendering.

**b. which part of NeRF do you think is the most important**

I think the Volume Rendering part is the most important. Because the rendering process should be differentiable so that we can calculate loss and apply back propagation to the network.

Also splitting the network into fine an coarse is pretty interesting. We can then focus on areas where object exists, which hugely improve the output quality.

**c. compare NeRF's pros/cons w.r.t. other novel view synthesis work**

- Pros:
  - NeRF models are self supervised. They can be trained using only multi-view images of a scene.
    Unlike many other 3D neural representation or view synthesis methods, NeRF models require only images and poses to learn a scene, and do not require 3D/depth supervision.
    The poses can also be estimated using Structure from Motion (SfM) packages such as COLMAP , as was done in certain scenes in the original NeRF paper
  - NeRF models are photo-realistic.
  - Require small space
- Cons:
  - Long training time
  - Inefficiency in rendering new views

**(10%) Describe the implementation details of Direct Voxel Grid Optimization(DVGO) for the given dataset. You need to explain DVGO's method in your own ways.**

It's basically trying to solve the problem of long training time of NeRF.
Instead of using MLP to implicitly learn the mapping from 3D point and view direction to colors and densities, they use a dense voxel grid to directly model the 3D geometry (volume density). Colors mapping is not the main scope of DVGO.
The benefit of this is super fast convergence but need to compensate for the problem of prone to suboptimal solutions.
This is the basic idea of DVGO. Others are details like Sharp decision boundary via post-activation, Search the coarse geometry of a scene, and reconstruct the fine detail including view-dependent effects.

**(15%) Given novel view camera pose from transforms_val.json, your model should render novel view images. Please evaluate your generated images and ground truth images with the following three metrics (mentioned in the NeRF paper). Try to use at least two different hyperparameter settings and discuss/analyze the results.**
**You also need to explain the meaning of these metrics.**

| SETTING | PSNR | SSIM | LPIPS |
|---|---|---|---|
| **SETTING 1 (COARSE STAGE 10000 ITER FINE STAGE 40000 ITER)** | 35.245 | 0.974 | 0.021 |
| **SETTING 2 (COARSE STAGE 5000 ITER FINE STAGE 20000 ITER)** | 35.127 | 0.971 | 0.023 |

Increasing the training iterations improve the performance slightly!

Meaning of different metrics:
**PSNR:**
peak signal-to-noise ratio. Used as a quality measurement between the original and a compressed image. Like MSE, but the MSE represents the cumulative

squared error while PSNR represents a measure of peak error.

**SSIM:**

Structural index similarity. A method for evaluating the similarity between two images. When one is original and one is compressed, it can be viewed as the indicator of image quality.

**LPIPS:**

Learned perceptual image patch similarity. Used to judge the perceptual similarity between two images. Since it’s calculating the distance between two outputs, the lower the better!

# Problem 2: Self-Supervised Pre-training for Image Classification (50%)

1. **(10%) Describe the implementation details of your SSL method for pre-training the ResNet50 backbone. (Include but not limited to the name of the SSL method you used, data augmentation for SSL, learning rate schedule, optimizer, and batch size setting for this pre-training phase)**

   Ans: I use the BYOL repo for pretraining. For augmentation, I use the same settings in the paper:

   In the paper, they seem to assure that one of the augmentations have a higher gaussian blur probability than the other. You can also adjust this to your heart's delight.

   ```
   augment_fn = nn.Sequential(
       kornia.augmentation.RandomHorizontalFlip()
   )

   augment_fn2 = nn.Sequential(
       kornia.augmentation.RandomHorizontalFlip(),
       kornia.filters.GaussianBlur2d((3, 3), (1.5, 1.5))
   )

   learner = BYOL(
       resnet,
       image_size = 256,
       hidden_layer = -2,
       augment_fn = augment_fn,
       augment_fn2 = augment_fn2,
   )
   ```

   I use random horizontal flip, and gaussian blur for one of the model (it seems that they use two models for both online and target).
   For lr and opt, I use 3e-4 and Adam. Batch size is 256.

**(20%) Please conduct the Image classification on Office-Home dataset as the downstream task. Also, please complete the following Table, which contains different image classification setting, and discuss/analyze the results.**

| Settings | Acc |
|---|---|
| A | 0.003 |
| B | 0.312 |
| C | 0.403 |
| D | 0.310 |
| E | 0.379 |

We can see that generally training the whole model will gain better result, compared with only train the classifier.

And training the whole model from scratch perform the poorest without doubt.

Also, I noticed that TA's backbone doesn't perform better than mine. I think this is because TA's backbone is trained on supervised data that has limited number, while my backbone is trained on way more pictures than TA's using SSL.