hw4_report

陳敬和 R11922166

Problem 1: 3D Novel View Synthesis

1. Explain

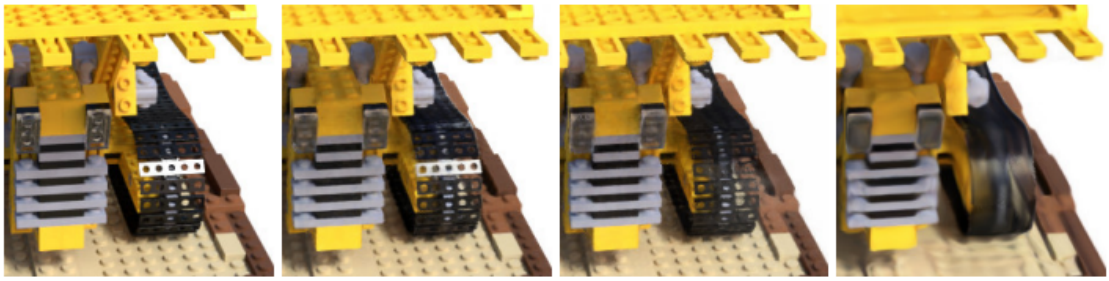
a. The NeRF idea in your own words

NeRF的核心精神是將物體與場景的資訊，encode到MLP中，再使用volume rendering將MLP的資訊投影出來，就可以對物體與場景render出連續且不存在於原始資料的視角。NeRF網路的輸入是一組5D的參數，分別是位置**(x, y, z)**與視角**(θ, φ)**，而輸出為顏色以及volume density，分別代表從某個視角看這個位置的顏色，與射線會停在此位置的機率(transparency)。

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

我認為是positional encoding。論文中提到，MLP在學習high frequency function的能力是比較差的，透過positional encoding可以在進入網路前，將輸入投影到高維空間，encoding function的公式如圖所示。而實驗結果可以看到下面那張圖，可以看到在使用positional encoding的時候，圖片的效果增強許多。

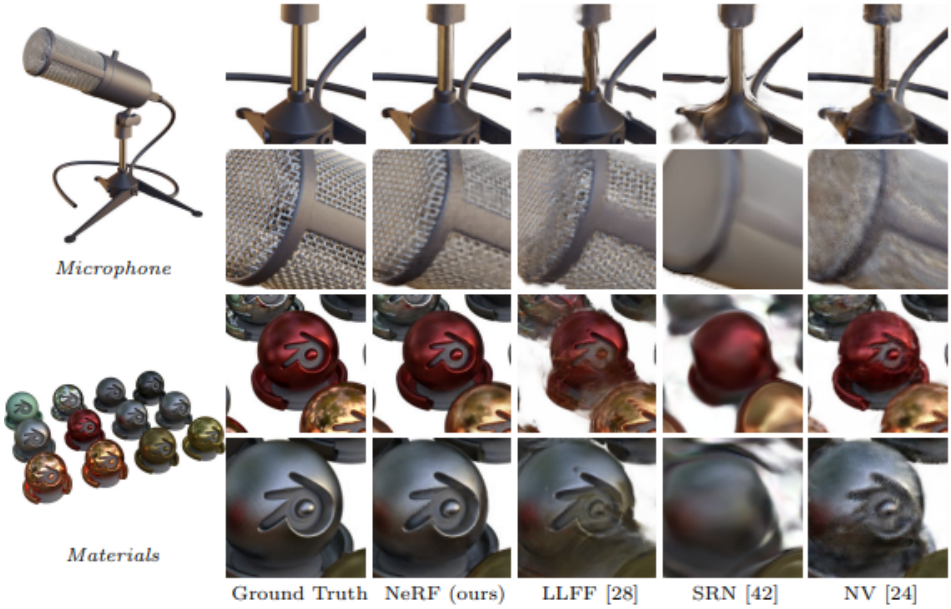
$$\gamma(p) = \left(\sin\left(2^0\pi p\right), \cos\left(2^0\pi p\right), \cdots, \sin\left(2^{L-1}\pi p\right), \cos\left(2^{L-1}\pi p\right)\right).$$



Ground Truth　Complete Model　No View Dependence　No Positional Encoding

c. Compare NeRF’s pros/cons w.r.t. other novel view synthesis work

以SRNs(Scene Representative Networks)來與Nerf做比較。SRNs可以將scene represent成一個連續可以微分的函數，將3D座標投影到基於feature的特徵表徵，實作上是將環境表示成不透明的表面，然後用 LSTM 的方式作 ray-marching 得到特徵值，最後再由特徵值轉換成影像的像素值。SRNs可以進行end-to-end的訓練，且只需要利用2D的圖片與相機位置就能夠學習，與當時的SOTA比較明顯優於其他競爭者。但SRNs有個缺點是將環境表示成不透明的表面，現實中的物體並不是都不透明的，從NeRF的Experiment中也可以看到，在一些擁有比較複雜表面的物體，SRNs沒有辦法清楚完整描述圖片的細節。



相比之下，NeRF在許多task上都能有很好的表現。然而NeRF的也有一些問題，針對每個場景都需要訓練一個網路，且訓練的時間也相當久，無法在real time上做出很好的應用，且拍攝的方式也有一定的限制。後續也有許多對NeRF的限制以及缺點進行改進的論文，除了第二題的DVGO加快訓練的時間，還有以few-shot方式進行訓練、強化NeRF在不同光影條件下的運行等等論文，相信在未來會有更多以NeRF為主題的題目出現。

2. 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

DVGO的目標是加速NeRF的訓練，主要透過幾個方法實現，第一使用兩種方式來避免進行direct voxel density優化時發生local minima，第二是提出post-activate voxel-grid interpolation，可以在low grid resolution的時候實現清晰的邊界建模。透過DVGO，可以將NeRF的訓練時間大幅減少至半小時以內，且有著不錯的效果。訓練快的原因有以下幾點，透過coarse to fine的方式跳過了大量無關點的採樣，以及將整個場景的訊息儲存在兩個voxel grid中，一個空間中的點只需要使用三個線性的插值就能得到，而不用過一個MLP。

實作方面，除了iteration次數以及voxel數量，其他的參數都與 https://github.com/sunset1995/DirectVoxGO一樣，詳細的數據可以見第三題。而在inference方面需要修改，要提前儲存transforms.json的filename，且更改儲存image的方式。在trace code之後，發現json是在load_blender.py做處理，檔名就在這邊與image、poses等一同處理；而dump_image這個參數原本是判斷要不要儲存圖片，這邊則修改成儲存圖片的path，並且對相對應的function做處理。

3. 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. Try to use at least two different hyperparameter settings and discuss/analyze the results.

- PSNR(Peak Signal-to-Noise Ratio)，通常用來衡量兩張圖片之間質量好壞，公式如下：

$$PSNR = 10 * log(MAX^2/MSE)$$

其中MSE為兩張圖片的mean square error，MAX則是圖像像素所能取得最大值。PSNR最小值為0，越大代表兩張圖片差異越小。

- SSIM(Structural similarity)，同樣的也是一種衡量兩張圖片相似程度的指標，基於人眼會提取圖片中結構化訊息的假設，能更符合人眼對影像品質的判斷。公式如下:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}$$

其中，$l(x,y)$比較x和y的亮度，$l(x,y)$比較x和y的對比度，$s(x,y)$比較x和y的結構，$\alpha, \beta, \gamma > 0$。SSIM最大值為1，數字越高代表兩張圖片的相似度愈高。

- LPIPS(Learned Perceptual Image Patch Similarity)，是在CVPR2018發表的新方法，使用deep feature來衡量圖片之間的相似度。文章中也有比較與傳統指標如L2/PSNR/SSIM等之間的差異，傳統的無法有效的解釋人類的感知，相比之下基於ML的感知相似度衡量更符合人類的認知。在這次作業使用vgg與AlexNet作為提取特徵的network architecture，因為是評估兩張圖片特徵的距離，因此LPIPS越小代表越好。
- Setting
  1. Setting 1: 將iteration number /= 2
  2. Setting 2: 將number of voxels /= 5
  3. Best: 將iteration number加上10000，且number of voxels *= 2

	PSNR↑	SSIM↑	LPIPS (vgg)↓	LPIPS(alex)↓
Default	35.174	0.9745	0.0415	0.0225
Setting 1	35.051	0.9734	0.0449	0.025
Setting 2	34.314	0.968	0.0536	0.0356
Best	35.376	0.9754	0.0387	0.0196

- Discussion

  可以發現在Setting 1將iteration次數減少，在PSNR與SSIM都有下降，而在LPIPS的部分則升高，因此可以推斷iteration number與performance有正相關，iteration次數越多，performance越好。

  而在Setting 2可以看到在voxel數量減少時，PSNR與SSIM下降許多，LPIPS也上升許多，可以推論voxel數量越多，performance也越好。

  綜合上面兩個觀察，Best的設置是將iteration次數加上10000，且將voxel數目加倍，即做出最好的結果。但在執行時間的部分確實有因為上述調整而有變長，要衡量執行時間與model performance之間的tradeoff。

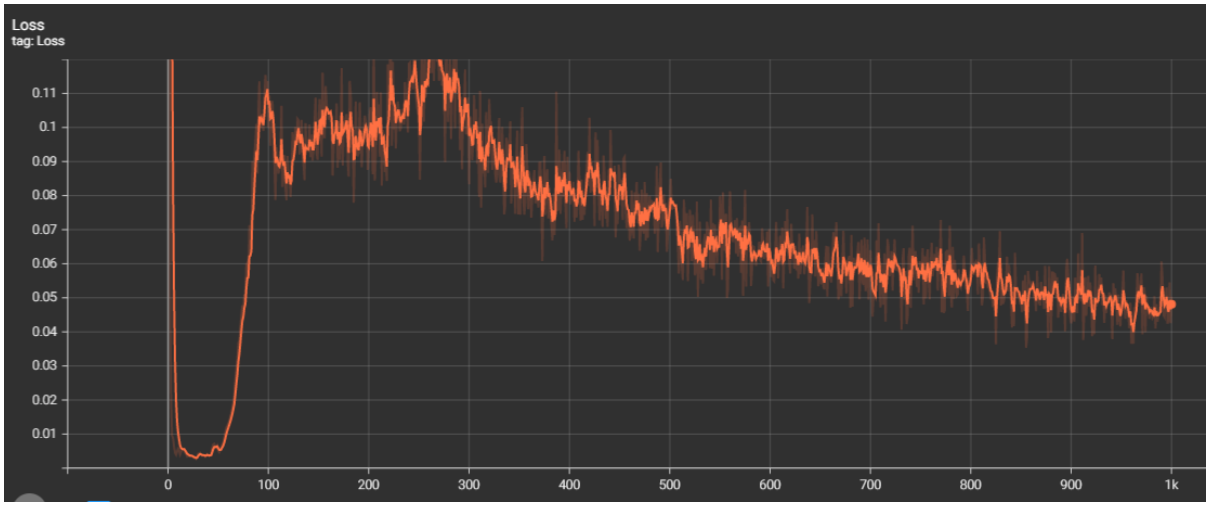
Problem 2: Self-Supervised Pre-training for Image Classification

1. Describe the implementation details of your SSL method for pre-training the ResNet50 backbone.

在這裡使用BYOL作為pre-training的方法，參照https://github.com/lucidrains/byol-pytorch的Readme設定，以ResNet50作為backbone，其餘設定如下:

Epoch	image size	Learning rate	batch size	optimizer	learning rate schedule
1000	128*128	3e-2	512	Adam	MutiStepLR w/ gamma = 0.5

使用A6000進行training，訓練時間大概為24小時。訓練初期在觀察BYOL的loss時(見下圖)，發現降的速度很快，但是在50幾個epoch的時候就回升，然後才穩定慢慢下降。一開始儲存checkpoint的方法是loss最小時存起來，但把loss最小的checkpoint拿去做downstream task時的表現慘不忍睹，把越後面epoch拿去作表現反而更好。不確定這邊BYOL的出來的loss是怎麼計算的，在訓練過程中會先下降上升在下降。但是在實驗中發現train的越久backbone的效果越好，因此最後訓練1000個epoch，接近24小時的時間後獲得還不錯的表現。若能訓練得更久，相信會有更好的downstream task表現。



2. 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.

Setting	Pre-training	Fine-tuning	Validation accuracy
A	-	train Backbone + classifier	33.25%
B	w/ label	train Backbone + classifier	38.67%
C	w/o label	train Backbone + classifier	49.26%
D	w/ label	Fix backbone, train classifier	34.98%
E	w/o label	Fix backbone, train classifier	40.39%

- Discussion

  SettingA是沒有pretrain，直接使用ResNet50作為backbone接上classifier進行訓練，在與其他組比較可以發現Setting A的accuracy是最低的，推論如果有先進行pretraining會對整個model的表現會有相當的幫助。

  接著先比較Pre-training階段有無使用label的Setting，BC與DE，可以無論是哪組在沒有label進行pretraining的表現較佳，個人認為是因為在資料有label的狀態下可能會限制model學習其他種task的能力，因此在downstream task上的accuracy沒有label的Setting表現反而會比有label的還好。

再來比較有沒有fix backbone的Setting，BD與CE，在backbone被fix的情況下，accuracy有一定的落差。可以推論在Fine-tune時，backbone還是有訓練的需要，只訓練一個classifier似乎無法很好的學習到downstream task。

Reference

https://github.com/lucidrains/byol-pytorch

https://github.com/sunset1995/DirectVoxGO