DLCV Hw4 Report

R10942198 林仲偉

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

1. (5%) Please explain: The NeRF idea in your own words / which part of NeRF do you think is the most important / compare NeRF's pros/cons w.r.t. other novel view synthesis work

Idea:

NeRF 簡單來說就是輸入目標體素的5D座標$(x, y, z, \theta, \varphi)$，，利用類神經網路直接預測出該體素的 4D 輸出$(r, g, b, \sigma)$. 其中$(x, y, z)$為視圖以焦距為原點之射線在立方體裡的位置，我們稱之為體素 voxel， (θ, φ)為從兩個不同焦距方向的視圖觀測該體素的角度， (r, g, b) 是該體素的紅綠藍數值，σ是該體素位置的體積密度(式1a)(式1b)。

最後通過傳統方法做渲染(式2a)(式2b)(式2c)，把這些體素生成圖片 ，再計算與正確圖片的均方誤差(式3)。

Novelty:

- 由於傳統渲染方法可以求導數，使得以上流程可以簡單地設計損失函數，並用類神經網路學習。
- 使用一種全新的編碼方式。由於該5D座標是離散的，故可以很好地表示邊緣很多的高頻場景

Compare:

- Pro: 傳統 3D 的視角合成方法（例如網格法、點雲法）都是需要一張中間場景進行建模，但由於中間場景仍然是一張離散分佈的圖片，所以合成的目標物會不夠精細（例如有鋸齒、偽影）。NERF 則是借用類神經網路之連續分佈的 latent feature 來表示目標物，所以可以更精細的合成出圖片 。
- Con: 有效像素太少，視圖中很多像素都不是目標物，跟 DVGO 比起來訓練費時。
- Ref: https://blog.csdn.net/minstyrain/article/details/123858806

$$(\sigma, \boldsymbol{e}) = \mathrm{MLP}^{(\mathrm{pos})}(\boldsymbol{x}) \,, \tag{1a}$$

$$\boldsymbol{c} = \mathrm{MLP}^{(\mathrm{rgb})}(\boldsymbol{e}, \boldsymbol{d}) \,, \tag{1b}$$

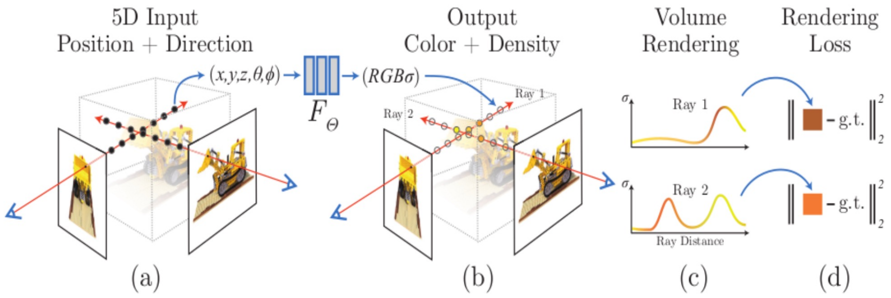
$$\hat{C}(\boldsymbol{r}) = \left(\sum_{i=1}^{K} T_i \alpha_i \boldsymbol{c}_i \right) + T_{K+1} \boldsymbol{c}_{\mathrm{bg}} \,, \tag{2a}$$

$$\alpha_i = \mathrm{alpha}(\sigma_i, \delta_i) = 1 - \exp(-\sigma_i \delta_i) \,, \tag{2b}$$

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j) \,, \tag{2c}$$

$$\mathcal{L}_{\mathrm{photo}} = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \left\| \hat{C}(\boldsymbol{r}) - C(\boldsymbol{r}) \right\|_2^2 \,, \tag{3}$$

Ref: DVGO https://arxiv.org/pdf/2111.11215.pdf



Ref: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

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

2. (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.

DVGO: 基於 NERF 對於 3D 場景重建的模型，DVGO 改良了三個地方：

$$\sigma = \mathrm{softplus}(\ddot{\sigma}) = \log(1 + \exp(\ddot{\sigma} + b)) \,, \qquad (5)$$

$$b = \log\left(\left(1 - \alpha^{(\mathrm{init})(c)}\right)^{-\frac{1}{s^{(c)}}} - 1\right) \,, \qquad (9)$$

1. Low-density initialization:

在 MLP 的計算中，使用算式(9) 作為初始化算式(5) b 的數值，在實驗及理論上皆獲得證實有效幫助模型更快收斂。

2. View-count-based learning rate:

由於實際上，某些體素在不同視角中可能很少出現（例如被遮住）。為了被遮住的表面可以很好的在多個視角還原，所以需要對不同的網格點設不同的 learning rate：紀錄那個網格點總共出現在幾個視角（假設有 n_j 個）還有總共有幾個視角（假設有 n_{max} 個）再把基礎的 learning rate 乘以該比率 $\frac{n_j}{n_{max}}$。

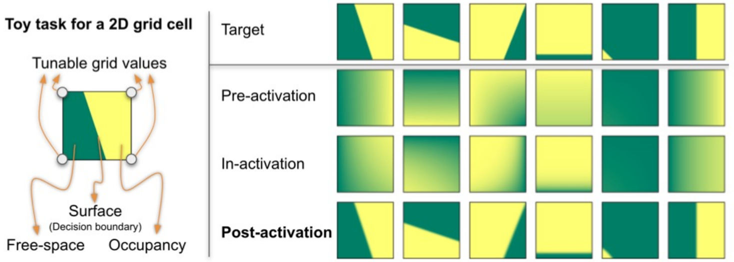
3. Sharp decision boundary via post-activated density voxel grid:

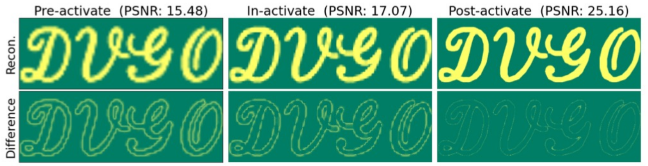
先對 voxel density value 做插值，再算 activation function (6c)，以產生清晰的邊界。這樣可以在更低的網格分辨率下，仍然產生清晰的建模。

$$\alpha^{(\mathrm{pre})} = \mathrm{interp}\left(\boldsymbol{x}, \mathrm{alpha}\left(\mathrm{softplus}\left(\boldsymbol{V}^{(\mathrm{density})}\right)\right)\right) \,, \qquad (6a)$$

$$\alpha^{(\mathrm{in})} = \mathrm{alpha}\left(\mathrm{interp}\left(\boldsymbol{x}, \mathrm{softplus}\left(\boldsymbol{V}^{(\mathrm{density})}\right)\right)\right) \,, \qquad (6b)$$

$$\alpha^{(\mathrm{post})} = \mathrm{alpha}\left(\mathrm{softplus}\left(\mathrm{interp}\left(\boldsymbol{x}, \boldsymbol{V}^{(\mathrm{density})}\right)\right)\right) \,. \qquad (6c)$$





(a) Visual comparison of image fitting results under grid resolution $(H/5) \times (W/5)$. The first row is the results of pre-, in-, and post-activation. The second row is their per-pixel absolute difference to the target image.

Implementation details:

整個訓練流程分成取樣率低的 coarse training phase 和取樣率高的 fine training phase。

- **Coarse training phase: 訓練 5000 iterations**

由於一般而言，大部分視圖中包含了很多非目標物的像素，所以我們需要初步的找到目標物的粗略 3D 區域，之後的 fine training phase 再對這些區域做進一步計算，如此就能減少視圖的每條射線需要取樣的點數。

這個步驟不需要太過精細的體素，Number of voxels 是 1024000，並且所有點的 (r, g, b, σ) 都只透過差值來還原 (式7a)(式7b)。只要找到不同訓練視圖視角圓錐所包含目標物的 Bounding Box 即可，如圖(c)。

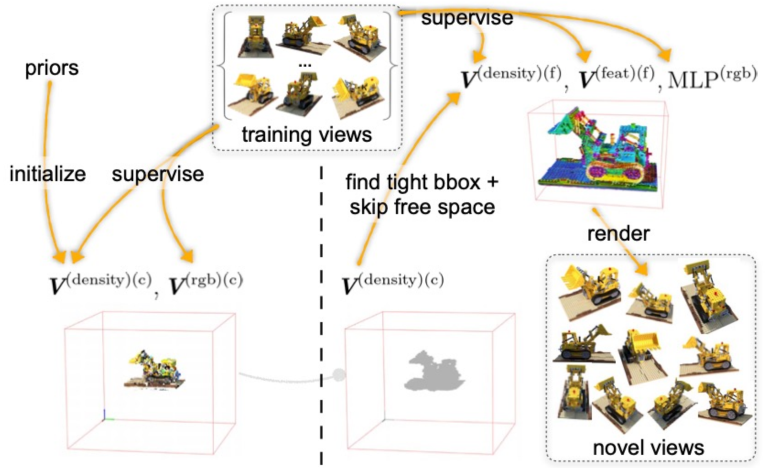
- **Fine training phase: 訓練 20000 iterations**

把目標物件的 Bounding Box 做進一步計算。

這個步驟需要精細的體素，Number of voxels 是 4096000。所有點的 (r, g, b, σ) 透過差值以及前面提到的 post-activation function 來還原 (式10a)(式10b)。MLP_{Θ} 一個 dimension 為 128 的類神經網路。

設置一個threshold τ^c，若 post-activation alpha value 低於 τ^c ，代表該點是在空間空間（已知沒東西的 3D 位置），否則代表是在未知空間（還不知道有沒有東西的 3D 位置）。取樣時跳過這些已知的空間空間，並且找到一個精細的 Bounding Box 去包住這些未知空間。

取樣方式則是使用 NSVF[1] 提到的 Progressive scaling：起始 Number of voxels = 4096000/pg_{ckpt} ，當訓練了 pg_{ckpt} 個iteration時，才再把 Number of voxels 翻倍，並用 trilinear interpolation 去更新 voxel grid 的 density 和 color.



(c) Coarse geometry searching (Sec. 5.1)

(d) Fine detail reconstruction (Sec. 5.2)

$$\ddot{\sigma}^{(c)} = \operatorname{interp}\left(\boldsymbol{x}, \boldsymbol{V}^{(\text{density})(c)}\right), \quad (7a)$$

$$\boldsymbol{c}^{(c)} = \operatorname{interp}\left(\boldsymbol{x}, \boldsymbol{V}^{(\text{rgb})(c)}\right), \quad (7b)$$

$$\ddot{\sigma}^{(f)} = \operatorname{interp}\left(\boldsymbol{x}, \boldsymbol{V}^{(\text{density})(f)}\right), \quad (10a)$$

$$\boldsymbol{c}^{(f)} = \text{MLP}^{(\text{rgb})}_{\Theta}\left(\operatorname{interp}(\boldsymbol{x}, \boldsymbol{V}^{(\text{feat})(f)}), \boldsymbol{x}, \boldsymbol{d}\right), \quad (10b)$$

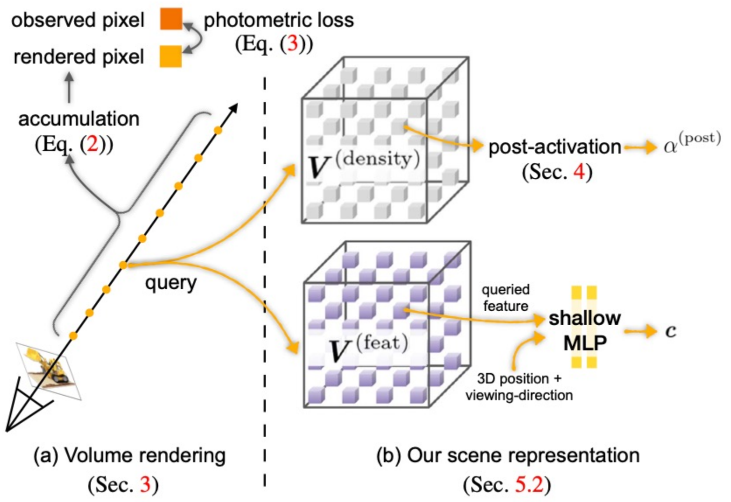
Training Setting	Coarse	Fine
iterations	5000	20000
Number of voxels	1024000	4096000
Stepsize	0.5	0.5
alpha_init	1e-6	1e-2
fast_color_threshold	1e-7	1e-4

[1] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020. 1,2,3,5,6,7,11,12,13,15,16,17

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

- (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.

Setting		PSNR	SSIM	LPIPS(vgg)
Step size	Fine voxel number			
0.5	**180^3**	**35.180**	**0.974**	**0.041**
2.0	180^3	32.386	0.956	0.072
4.0	180^3	28.413	0.922	0.112
0.5	120^3	34.870	0.972	0.046
0.5	80^3	34.059	0.968	0.055



Ref: DVGO https://arxiv.org/pdf/2111.11215.pdf

Step size 代表的是重構 3D 體素時，從視圖的焦距所發出的射線之 query point 單位距離。如圖(a)中射線上的那些點彼此的距離。Fine voxel number 代表的是生成體素的數目。如圖(b)中正方體內部的那些點的總數。

可以發現當 step size 越大/ voxel number 越小，PSNR和SSIM皆下降，LPIPS則微幅上升，代表圖片與原圖差異變大。
推論是 step size 越大，從視圖取樣的像素點越少，所以最後生成的體素就越不精細。
而 voxel number 越少，代表生成的 3D 模型解析度越低，所以最後生成的體素也會越不精細。

Metric explanation：下一頁

- **PSNR: Peak signal-to-noise ratio**

$$\mathbf{PSNR} = \mathbf{10}\ \boldsymbol{log}\frac{\boldsymbol{MAX^2}}{\boldsymbol{MSE}}$$

衡量一個訊號最大功率和雜訊功率的比值。 MAX為訊號的最大值（例如 8bit 圖像為 255）。 MSE 為生成訊號與原訊號的均方誤差平均。 **PSNR 越高，代表訊號的重建程度越好。**

- **SSIM: Structural Similarity**

$$\mathbf{SSIM(x,y)} = \frac{\left(\mathbf{2}\boldsymbol{\mu_x\mu_y} + \boldsymbol{C}\mathbf{1}\right)\left(\mathbf{2}\boldsymbol{\sigma_{xy}} + \boldsymbol{C}\mathbf{2}\right)}{\left(\boldsymbol{\mu_x^2} + \boldsymbol{\mu_y^2} + \boldsymbol{C}\mathbf{1}\right)\left(\boldsymbol{\sigma_x^2} + \boldsymbol{\sigma_y^2} + \boldsymbol{C}\mathbf{2}\right)}\boldsymbol{\epsilon}[\mathbf{0},\mathbf{1}]$$

PSNR只能判斷生成訊號和原本訊號每個點的平均距離，但無法衡量生成訊號結構上是否跟原本訊號相似。

SSIM 則是將訊號分成亮度Ｉ、對比度Ｃ、結構Ｓ這三種不同屬性，計算這三個變數的共變異數之後，再相乘所得到的指標。 **SSIM 越高，代表兩個訊號越相似。** SSIM = 1 代表兩個訊號完全相同。

（註： C1、C2為非零常數，避免分母等於0）

- **LPIPS:**

PSNR、SSIM 只能評估結構相似性，但不能很好的解釋人類對圖像的感知，且對模糊圖像不敏感。LPIPS則是藉由基於人類根據原圖和失真圖進行評分得到d0，收集這些標註當作資料集。再把原圖和失真圖丟到一個特徵提取網路 CNN（例如 vgg16 or alexnet），把每一層特徵算L2距離取平均得到d1，最後訓練一個DNN，去學習如何最小化 d0 和 d1 的差距。**LPIPS 越小，代表兩個圖片的越像。**

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)

- **Epoch**: Train for 1000 epochs (roughly one day)
- **SSL method**: BYOL
- **Data augmentation for SSL**: Resize to 128, normalize by mean=[0.485, 0.456, 0.406] and std=[0.229, 0.224, 0.225]
- **Learning rate schedule**: StepLR. Initial learning rate 0.05, decay 0.2% for every 100 steps.
- **Optimizer**: SGD
- **batch size**: 64

No other trick is used.

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

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

- (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.

Setting	Pre-training (Mini-ImageNet)	Fine-tuning (Office-Home dataset)	Validation accuracy (Office-Home dataset)
A	-	Train full model (backbone + classifier)	0.162
B	w/ label (TAs have provided this backbone)	Train full model (backbone + classifier)	0.229
***C**	***w/o label (Your SSL pre-trained backbone)**	***Train full model (backbone + classifier)**	***0.585**
D	w/ label (TAs have provided this backbone)	Fix the backbone. Train classifier only	0.201
E	w/o label (Your SSL pre-trained backbone)	Fix the backbone. Train classifier only	0.445

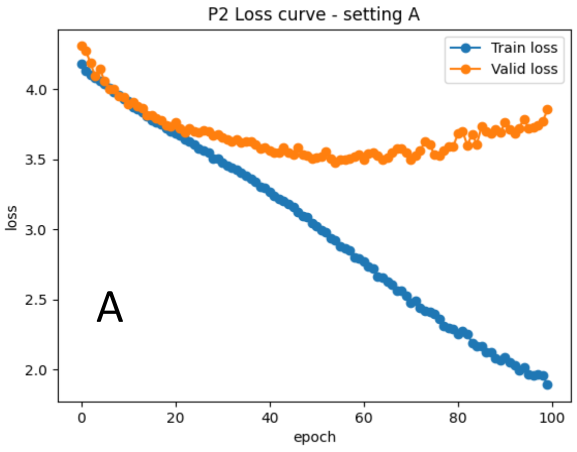
Config	value
Optimizer	SGD
Learning rate	3e-4
Scheduler	StepLR, gamma=0.998
Batch size	16
Epoch	100

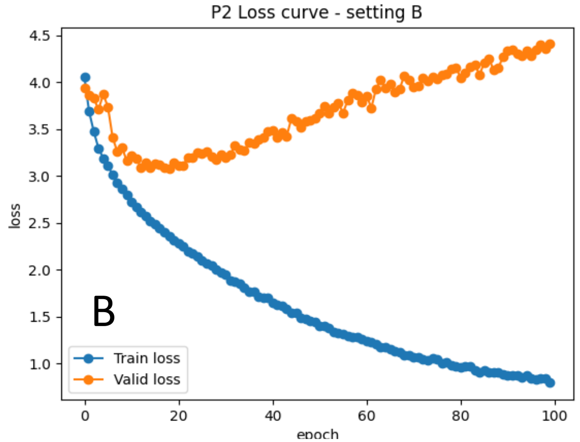
We found that using SSL pre-trained backbone to train the full model can achieve the best result among all of the settings. Additionally, using SSL pre-trained backbone is better than using supervised learning pre-trained backbone regardless whether the classifier layer is fixed or not.

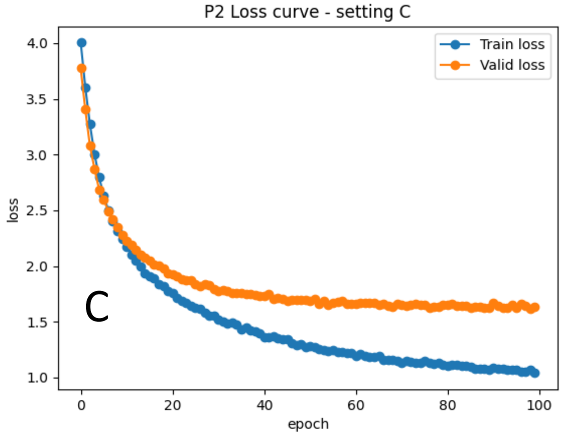
另外，由於觀察發現 downstream task dataset 單張差異過大，所以我覺得 downstream task batch size 不能調大大。

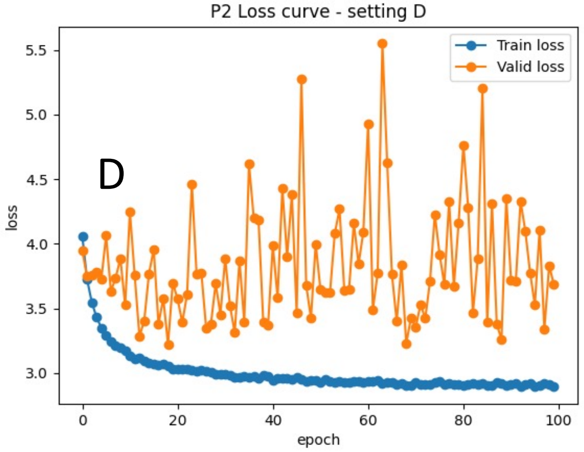
下一頁還有

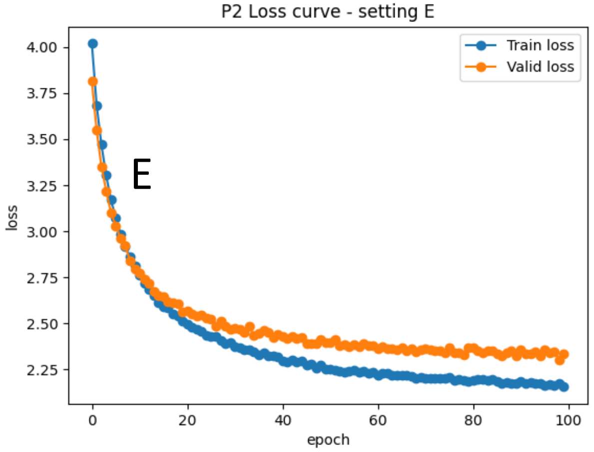
Setting	Pre-training (Mini-ImageNet)	Fine-tuning (Office-Home dataset)
A	-	Train full model (backbone + classifier)
B	w/ label (TAs have provided this backbone)	Train full model (backbone + classifier)
***C**	***w/o label (Your SSL pre-trained backbone)**	***Train full model (backbone + classifier)**
D	w/ label (TAs have provided this backbone)	Fix the backbone. Train classifier only
E	w/o label (Your SSL pre-trained backbone)	Fix the backbone. Train classifier only











These figures show the loss-epoch plot of training and validation for each setting. The model of setting A, B, D overfit in the early training stage. Setting C and E achieve better performance and have more stable training procedure, which implies using SSL as pre-trained method can get more desired result.

Note that setting D is not stable in comparison to setting B. That’s because the class number of downstream task and pre-train task are different. The pretrain backbone does not fit well in the downstream task of setting D. On the other hand, using SSL pre-trained backbone can alleviate the problem since the backbone model is trained without knowing the belonging class and/or number of total classes