HW4 Report

B08901165 電機四 南策昇

Problem 1

1. (a)NeRF將一個景象各個角度照片的每個點轉換成空間中那個點的五維向量，其中前三維是該點xyz，後兩維是視角theta、phi。之所以要多上視角的資訊是因為視角不同也會影響物體的遮蔽情形和顏色。之後，作者要求model能夠預測出空間中點的density和r、g、b共四個數值。因為rgb數值groundtruth可以從照片上取得，因此這個model在邏輯上是可以取得不錯效果的。

除了這樣直觀的設計外，作者還加入幾個改進。首先，這個model的inputs並不是同時進入的，outputs也不是同時出來，xyz首先會經過一堆MLP，產生一個feature vector和density，feature vector再和theta phi做連接，過完後面的model產生rgb。之所以這樣做是因為density理論上的確與看物體的視角無關，因此這樣可以幫助model學到更有用的資訊。再來，作者使用position encoding將位置資訊投影到更高維度的空間，幫助model學習。此外，如果要重建模型，原本空間中的每一個點都需要去判定他的透明度，為了加快model速度，所以作者先用一個coarse model快速判定哪些點密度較高，有機會影響輸出，接著再用fine model依據coarse model的結果，將較有機會影響輸出的點的採樣機率調高，這樣就省去採樣那些不重要的點的時間了。

1. (b)我認為最重要的是將theta和phi列為inputs，這也是NeRF的核心之一。作者做的其他改進大多是讓model比較容易train，但是增加角度卻是直接提供給model更多資訊，並且僅用在預測rgb上面也能幫助到model從xyz找出density。可以看到原始paper中有做ablation study，少掉view dependency的版本也是各項metrics差距最大的。
2. (c)NeRF的優點是畫質很好，而且因為有得出詳細的density等資訊的關係，可以應用在其他相關的任務上。但是他最大的缺點就是運算實在太慢了。相比DVGO這篇paper只需要用到十幾分鐘，NeRF卻要用到十幾二十小時。
3. DVGO的架構大致上和NeRF差不多，但是作者做了一些改進。首先，作者使用voxel grid來表示density，這樣的話就可以很快地用插值法取得任意xyz的density。但此方法的缺點是最佳化不易，因此作者使用Mip-NeRF作為activation function，直接優化voxel grid，並提出post-activation的概念，在插值後才通過activation function，如此一來可以產生出更sharp的面，比較好用voxel grid來表示。這裡作者再讓voxel grid初始在接近0的數值，而且用sample到某一點的次數來動態調整learning rate，越常sample到learning rate越大，以上這些改進都是為了要更好的去optimize以voxel grid表示的方法。最後作者也用了coarse、fine兩個model分層產生結果，先用coarse model勾勒出大概的模樣，再用fine model去將整個畫面產生出來，這樣的設定同樣也加快了整體速度。DVGO每訓練一段時間就會增加voxel grid的數量，提高解析度，另外在訓練過程中，能省略的render步驟就省略，比方說這個視角下，某些光線並不會打到任何物體，或者光線通過空的區域，那其實都是可以不用sample的，就直接省略就好。也因此，DVGO才可以在基於NeRF的情況下，做出那麼大的進步。

|  |  |  |  |
| --- | --- | --- | --- |
|  | PSNR | SSIM | LPIPS |
| Setting 1 | 35.1823 | 0.9744 | 0.0412 |
| Setting 2 | 35.1897 | 0.9745 | 0.0413 |

Setting 1 is set with default hyperparamers in [sunset1995/DirectVoxGO: Direct voxel grid optimization for fast radiance field reconstruction. (github.com)](https://github.com/sunset1995/DirectVoxGO)

Setting 2 is set the same as setting 1 except the based learning rate is 5e-3.

PSNR：將圖片可能的最大像素值除以兩張圖片的MSE，然後以dB表示，所以可以看成一個常數減掉一個取對數的MSE值，也就是說PSNR越大越好。

SSIM：將luminance、contrast、structure這三個圖片要素透過各自的公式計算出兩張圖片的差異，通常都是取一個小patch計算，然後滑動patch在計算一次，滑完整張圖片取平均。兩張圖片的std和mean越接近，SSIM越大，因此SSIM也是越大越好。

LPIPS：將圖片通過一個model（我是使用VGG），並將兩張圖片在各層的output彼此拿來比較計算距離，然後再乘上該層的權重，經過簡單的normalize之後得出。由於是計算距離，所以越小越好。

可以發覺setting 1和setting 2結果是差不多的，我猜測可能和DVGO有動態調整各個voxel的learning rate 有關，所以些微的based learning rate差異並不會太影響到最終結果。不過還是可以觀察到setting 2稍微高一點，其實在setting 1的時候，train 完20000個epochs時並沒有收斂，可以看到PSNR仍在上升，所以如果把learning rate調高一點，應該是有機會讓他收斂更快的。雖然效果不明顯，但是仍然可以看出一點點端倪。

Problem 2

1. 我使用BYOL作為SSL的方法，在backbone的部分使用torchvision.models 提供的resnet50，並且只pretrain到avg\_pool layer，pretrain了100個epochs，optimizer使用Adam，learning rate 為3e-4，beta使用pytorch預設數值，batchsize 64，並且使用助教提供的preprocess方式做圖片處理，先resize到128，再center crop成128\*128，然後用ImageNet的標準做Normalization。

|  |  |  |  |
| --- | --- | --- | --- |
| Setting | Pre-training | Fine-tuning | Validation accuracy |
| A | - | Full model | 0.2980 |
| B | TA’s backbone | Full model | 0.3448 |
| C | SSL pretrained | Full model | 0.4064 |
| D | TA’s backbone | Fix backbone | 0.1970 |
| E | SSL pretrained | Fix backbone | 0.2512 |

Mean of A~E: 0.2995

觀察可以發現，以上結果大概可以分成三個區段(B, C)、(A)、(D, E)。可以看到有使用Full model做training的效果明顯較好，我想這也是因為我只使用了一層Linear layer而且沒有加任何的activation function，導致只有更新最後一層，model並不足以學到足夠的東西。另外，(B, C)較(A)好，顯示pretrain還是很重要的，不管有沒有使用label，至少都有幫助model去擷取到圖片中的信息。接著，C比B好，E比D好，顯示在這次作業中，不使用label pretrain才能達到比較好的效果，推測原因是因為office和home兩個資料集的分類問題並不相同，如果使用label，那model就會為了要學會home的分類而擷取那些只有在home才會出現的圖片信息，然而擷取這些信息卻在office上面毫無用武之地，甚至可誤導model。如果不使用label，那model學會的只是如何分析任一張圖片，並沒有學會要做什麼特定的任務，就像小孩一樣，在還沒有特定的生活模式的時候，要怎麼培養或者教育都很容易，但是如果已經長大，有了自己的生活習慣，要強迫他學習什麼就變得比較困難。