影像處理 HW3

109550087 單宇晟

這次作業我選擇實機練習的第一個: *E-NeRV: Expedite Neural Video Representation with Disentangled Spatial-Temporal Context* 來進行操作

- **論文介紹、原架構缺點及可改善之處**

NeRV (Neural Representation for Videos)是一種圖像隱式神經表示法，也就是上次作業實作的方法，是為了解決傳統 INR (Implicit Neural Representation)的一些限制才被設計出來的。然而，NeRV 還是有一些缺點，那就是隨著通道維度的增加，整個 model 的大小也會迅速增加，導致出現很多不必要的 parameter。因此，這篇 paper 提出了一種名為 E-NeRV (Expedite Neural Video Representation)的方法來改善這些缺點。

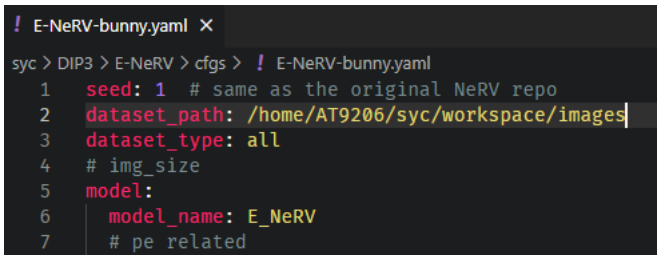
在 NeRV 的架構中，發現 MLP 的最後一層其實很冗長。為了解決這個問題，E-NeRV 提出了 spatial-temporal 資訊的 disentangled formulation。E-NeRV 並不像原本 NeRV 那樣，直接從 input 中生成 frame feature map，而是把 spatial 和 temporal 的 context 分開。接下來，為了把兩者重新結合，會進行 element-wise multiplication。如此一來，就可以知道 spatial-temporal representation。

另外，NeRV 因為做了 subsequent pixel-shuffle operation，導致 trainable weight 的數量的大幅增加。作者在這裡對 NeRV 的 block 進行一些調整，他們使用維度較小的兩個 consecutive convolution kernels 取代原本的 convolution kernel，並改在中間做 pixel-shuffle operation、加入 intermediate dimension。

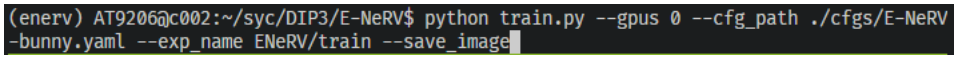
整體來說，NeRV 和 E-NeRV 都使用十分類似的架構，只是 E-NeRV 對特定部分做了修改，透過上面提到的 disentangling 和 fusion，還有對 NeRV 結構的改善，E-NeRV 減少了 NeRV 中的 redundancy，甚至提高了性能和收斂速度，也保持了較少的參數數量。至於未來的方向，作者打算把這個方法運用到其他下游的任務，像是 optical flow estimation 和 video super-resolution。

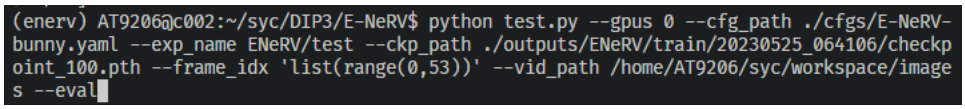
- **實機練習**

由於這次 github 的指示並沒有寫得像上次 NeRV 那篇清楚，我首先了解了作者 code 的大概架構，並將 data path 改成自己的資料(bunny.yaml 裡)，同時也調整成我想跑的 epoch 數量(100)，而最重要的資料集，我是用和 HW2 一樣的照片，方便進行比較。



接著，我參考網路上的 code 重新將 main.py 改寫，分成了 train.py 以及 test.py，剩下的部分則沒有更改，並按照指令輸入到 terminal 去執行。



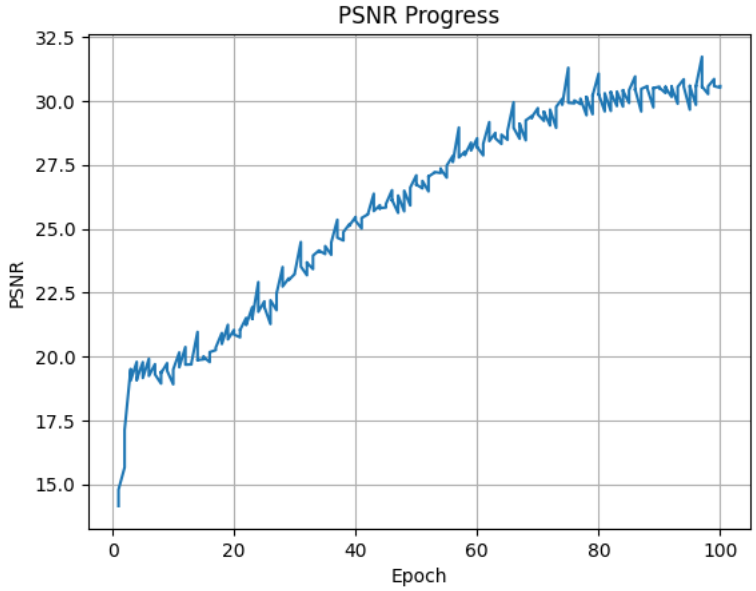


- **模型訓練與評估結果**

train

```
[2023-05-24 18:21:10] Rank:None, Eval, Step [53/53], PSNR: 39.02, MSSSIM: 0.9968
[2023-05-24 18:21:10] -> total time on evaluate:        4.89
[2023-05-24 18:21:10] ==> Eval: psnr_now: 39.02  psnr_best: 39.02        msssim_now: 0.9968     msssim_best: 0.9968

[2023-05-24 18:21:10] --- Total Training Time: 1:11:51.838572 ---
```



test

```
[2023-05-25 07:17:15] Frame #21/53 PSNR: 30.72, MSSSIM: 0.9821
[2023-05-25 07:17:15] Frame #22/53 PSNR: 30.86, MSSSIM: 0.9771
[2023-05-25 07:17:15] Frame #23/53 PSNR: 30.81, MSSSIM: 0.9767
[2023-05-25 07:17:15] Frame #24/53 PSNR: 32.07, MSSSIM: 0.9847
[2023-05-25 07:17:16] Frame #25/53 PSNR: 29.69, MSSSIM: 0.9744
[2023-05-25 07:17:16] Frame #26/53 PSNR: 29.85, MSSSIM: 0.9794
[2023-05-25 07:17:16] Frame #27/53 PSNR: 30.88, MSSSIM: 0.9765
[2023-05-25 07:17:16] Frame #28/53 PSNR: 30.13, MSSSIM: 0.9768
[2023-05-25 07:17:16] Frame #29/53 PSNR: 30.65, MSSSIM: 0.9750
[2023-05-25 07:17:16] Frame #30/53 PSNR: 30.61, MSSSIM: 0.9807
[2023-05-25 07:17:16] Frame #31/53 PSNR: 30.70, MSSSIM: 0.9817
[2023-05-25 07:17:17] Frame #32/53 PSNR: 30.85, MSSSIM: 0.9794
[2023-05-25 07:17:17] Frame #33/53 PSNR: 31.73, MSSSIM: 0.9826
[2023-05-25 07:17:17] Frame #34/53 PSNR: 30.12, MSSSIM: 0.9760
[2023-05-25 07:17:17] Frame #35/53 PSNR: 29.94, MSSSIM: 0.9807
[2023-05-25 07:17:17] Frame #36/53 PSNR: 30.00, MSSSIM: 0.9745
[2023-05-25 07:17:17] Frame #37/53 PSNR: 30.52, MSSSIM: 0.9757
[2023-05-25 07:17:17] Frame #38/53 PSNR: 31.41, MSSSIM: 0.9809
[2023-05-25 07:17:17] Frame #39/53 PSNR: 31.02, MSSSIM: 0.9809
[2023-05-25 07:17:18] Frame #40/53 PSNR: 31.01, MSSSIM: 0.9800
[2023-05-25 07:17:18] Frame #41/53 PSNR: 30.01, MSSSIM: 0.9764
[2023-05-25 07:17:18] Frame #42/53 PSNR: 30.72, MSSSIM: 0.9814
[2023-05-25 07:17:18] Frame #43/53 PSNR: 30.78, MSSSIM: 0.9792
[2023-05-25 07:17:18] Frame #44/53 PSNR: 29.92, MSSSIM: 0.9734
[2023-05-25 07:17:18] Frame #45/53 PSNR: 29.91, MSSSIM: 0.9773
[2023-05-25 07:17:18] Frame #46/53 PSNR: 30.04, MSSSIM: 0.9751
[2023-05-25 07:17:19] Frame #47/53 PSNR: 31.11, MSSSIM: 0.9817
[2023-05-25 07:17:19] Frame #48/53 PSNR: 30.94, MSSSIM: 0.9793
[2023-05-25 07:17:19] Frame #49/53 PSNR: 30.97, MSSSIM: 0.9818
[2023-05-25 07:17:19] Frame #50/53 PSNR: 30.09, MSSSIM: 0.9772
[2023-05-25 07:17:19] Frame #51/53 PSNR: 30.18, MSSSIM: 0.9764
[2023-05-25 07:17:19] Frame #52/53 PSNR: 30.30, MSSSIM: 0.9752
[2023-05-25 07:17:19] Frame #53/53 PSNR: 30.07, MSSSIM: 0.9749
```

- **分析與討論**

透過 PSNR Progress 可以看到，在 training 的前幾個 epoch，psnr 上升的十分迅速，直到將近 epoch 100 才趨於穩定，足見此方法的有效性。另外，雖然這次的 epoch 數達到 100，但實際的訓練時間也不超過 2 小時，比起 NeRV 的訓練時間縮短了很多很多，上次作業 epoch 數量只有 20，但卻需要 4-6 個小時才能訓練完成，我覺得這跟最一開始 paper 裡提到的 NeRV 的缺陷有關，由於我所準備的圖片畫質較高，可能會造成在 training 的過程，input 的 dimension 提高，讓 NeRV model 整體的參數數量過大，訓練時長也才會那麼久，而 E-NeRV 則改善了這方面。

最後，雖然 E-NeRV 輸出的結果不是 gif，而是一張張圖片，但藉由跟上次的圖片比較，整體的清晰度甚至還有所上升，從 psnr 也可看出來，上次我的 PSNR 只到 22 左右，這次卻到了 30 上下。總體而言，我認為 E-NeRV 藉由 spatial 和 temporal 的運用，以及 block 的重新安排，對於原本設計的改善與提升是十分明顯的，也讓我了解到其實只要對神經網絡的架構、組成做一些微調，或是在 feature extraction 上運用不同的方法，就能對結果造成十分巨大的改變。

HW2 NeRV 結果

Epoch	10	15	20
PSNR (val)	22.3551	22.4218	22.3415

NeRV result

E-NeRV result