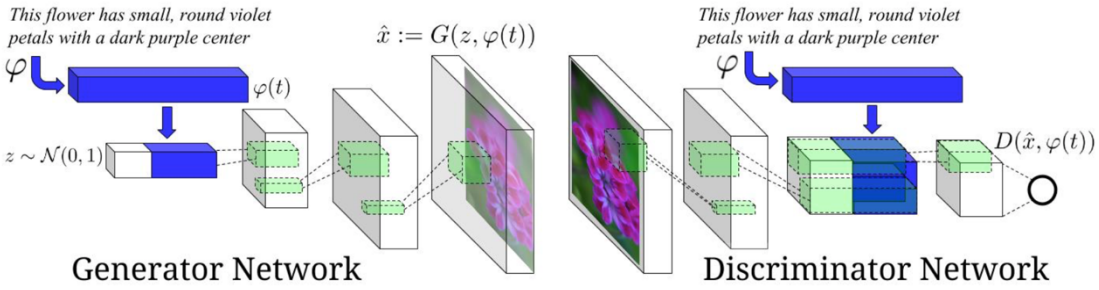
HW4

Model description

參考 Generative Adversarial Text to Image Synthesis（Scott et al. 2016）這篇 paper 提出的 model structure 與 training 流程，如下圖。



將 condition vector 與 noise vector 做 concat 後成為 generator 與 discriminator 的 input vector，而我是將頭髮與眼睛顏色變成總共 23 個類別並做 one hot encoding，若 condition 為紅髮綠眼，condition vector 在其對應的維度為 1 其餘為 0，再加上 23 維的 Gaussian noise（mean:0、std:0.2），而 discriminator 的 loss 分為四個部分，除了 input real image 與 real condition 要趨近 1、real image 與 wrong condition 要趨近 0、fake image 與 right condition 也要趨近 0，還多了 wrong image 與 real condition 要趨近 0，將這些的 sigmoid cross entropy 的 loss 加總。

Generator 的 input 為 23 維的 condition vector 接上 output 為 20 維的 Dense 並與 30 維的 noise vector 做 concat，並接上 output 為 64*8*(64/16)* (64/16)維的 Dense 並 reshape 到 (64/16), (64/16), 64*8，再來依序接上 64*4 個 filters、output 為 8x8，64*2 個 filters、output 為 16x16，64 個 filters、output 為 32x32，3 個 filters、output 為 64x64 的 deconvolution layers（stride 皆為(2,2)，kernel size 皆為 5x5）。

Discriminator 的 input 有 64x64x3 的 image 與 27 維的 condition vectors，首先將 image 通過 64、64*2、64*4、64*8 個 filters，stride 皆為 (2,2)，kernel size 皆為 5x5 的 convolution layers 最後輸出為 (4,4,512) 維，將 condition vector 接上 output 為 20 維的 Dense 並 reshape 到(1,1,20) 再做 tile 成 (4,4,20)，與 image 的輸出做 concat 再通過 64*8 個 filters、kernel size 為 1x1 且 stride 為 (1,1) 的 convolution layer，最後在接上 output 1 維的 Dense。

How do you improve your performance

將原本的 sentence to vector 改成只將 hair 與 eyes 的顏色共 23 個類別做 one hot encoding 作為 condition vector 的 input 會導致最後喪失隨機性，可能因為最後 input layer 的 weight 較偏重 condition vector，而 noise layer 的 weight 較少使得同一個 condition 不同 noise 的圖片都長得差不多並且較模糊不清，所以我將 one hot encoding 的 vector 再加上平均為 vector 的 mean、標準差為 vector 的 standard deviation 的 Gaussian noise，產生帶點 nosie 的 condition vector，如此一來可將 weight 的分布較為平均，即可解決喪失隨機性的問題。

參考的 paper 針對 discriminator 的 loss 只有 3 個部分，input real image 與 real condition、real image 與 wrong condition、fake image 與 right condition，我再多加了 wrong image 與 real condition 的比較，使得 condition 的效果增加許多。

因為助教提供的 tags 有很多圖片都沒有關於 hair 與 eyes，並且有同一張圖片的 tags 有多個頭髮的顏色，若是每個圖片皆取最多分數的 tag，則將不會滿足 23 個類別，所以我使用了 illustration2vec (i2v) 這個 model 將所有圖都餵進去產生關於 hair 與 eyes 的顏色，而當每個圖片的 tags 有缺少的就取 i2v 生成的，如此以來，所有圖片都有關於 hair 與 eyes 的 tags 了。

附上 illustration2vec (i2v)：https://github.com/rezoo/illustration2vec

Experiment setting and observation

在 training 的時候不是 train 的 epoch 越多效果越好，很容易 train 到後面圖片開始會越來越模糊且 condition 也沒有更加明顯，也沒有個絕對的標準來判定 overfitting 並自動停止 training，只能將每個 epoch 的 model 都存起來，並且最後觀察 generator 與 discriminator 的 loss 或是在每個 epoch 都 print 出 predict 的圖片來判斷 model 好壞，通常都是 train 到 80 幾個 epoch 生成的圖片是最清晰而 condition 的表現也最好。

Training 的時候，real image 與 real condition 要趨近 1 的 loss 相對於另外 3 個 loss，real image 與 wrong condition 趨近 0、fake image 與 right condition 趨近 0、wrong image 與 real condition 趨近 0，一直比這 3 個還要高出許多，並且下降的幅度也較小，導致當圖片的 condition 已經漸漸明顯時，圖片則會開始扭曲並模糊。 所以導致 train 越久的時候雖然 condition 很明顯，但圖片則是相當模糊。 為了解決這個狀況，有調整 loss 加總的權重，但還是沒有什麼顯著的成長，可能就是將 noise vector 與 condition vector 做 concat 的缺點吧。