[ADL—hw4 report]

- Model description(2%)

(Must include model strucuture, objective function for G and D)

[參數設定]

Batch size = 64 / learn rate = 2e-4 / noise dim = 100 / 使用 skip-thought vector

[簡述]

Generator 會根據給的文字來生成圖片，學習生成出能騙過 discriminator 的假圖片，目標是減少 g_loss，discriminator 則是用來判別圖片，學習找出 generator 生成的假圖片，並依此給出一個分數，目標是減少 d_loss，另外，還有一個 classifier 的作用和 generator 相反，是依據給的圖片來生成文字，學習真的圖片和真的 label 間的對應，以及生成和給的文字相對應的圖片，目標減少 c_loss。

[generator]

使用 cnn 模型，先用一層 fully_connected，inpout 為加上 noise 後的假 label，output 經過 reshape 後為(64,512,8,8)，接著用四層 conv2d_transpose，output shape 分別為(64,256,16,16)、(64,128,32,32)、(64,64,64,64)、(64,3,64,64)，最後 output 即是假的 image，其中 activation 都用 relu，stride 前三層為 2，最後一層為 1，初始化參數使用 random_normal_initializer(0, 0.04)，並用 batch_norm 做 normalize，把 fused 設為 True。

[discriminator]

使用 cnn 模型，首先用四層 conv2d，input 為 image(訓練資料中的真 image 或 generator 生成的假 image)，output shape 分別為(64,64,32,32)、(64,128,16,16)、(64,256,8,8)、(64,512,4,4)，其中 activation 都用 lrelu，stride 皆為 2，kernel_size 則為 3，並用 batch_norm 做 normalize，把 fused 設為 True，接著經過 reshape 攤平後再用一層 fully_connected，shape 變為(64,1)，最後 output 即為判別出的分數。

[classifier]

使用 cnn 模型，第一層為 conv2d，input 為 image(訓練資料中的真 image 或 generator 生成的假 image)，output shape 為(64,32,64,64)，kernel_size 為 5，stride 為 1，activation 用 relu，第二層為 max_pool2d，output shape 為(64,32,32,32)，stride 為 2，第三層為 conv2d，output shape 為(64,64,32,32)，kernel_size 為 5，stride 為 1，activation 用 relu，用 batch_norm 做 normalize，把 fused 設為 True，第四層為 max_pool2d，output shape 為(64,64,16,16)，stride 為 2，接著用 reshape 攤平，最後加上兩層 fully_connected，output shape 分別為(64,1024)、(64,4800)，最後的 output 為依據 input 的 image 生成的 label，其中第一個 fully_connected 用 relu 做 activation，第二個則無。

[objective function for G and D]

首先使用 discriminator 去對圖片做評分:

→disc_real = discriminator 對真 image 的評分

→disc_fake = discriminator 對假 image 的評分

接著計算 discriminator 和 generator 的 loss:

→d_loss = tf.reduce_mean(disc_fake - disc_real)

→g_loss = tf.reduce_mean(-disc_fake)

目標為最小化 loss:

都使用 RMSPropOptimizer 做 Optimizer，並對 d_loss 的梯度做 clip，限制值須在 -0.01~0.01 之間

- How do you improve your performance (2%)

  1. 計算 c_loss 的方式改為使用 softmax_cross_entropy_with_logits
  2. 原本使用一般 GAN，改為使用 WGAN

  原本的 classifier 會依據圖片生出文字，並且用 rnn 模型做 embedding，計算 loss_c 的方式就是去算真的 label 和假的 label 間的 cosine similarity，實際跑的時候，大概 50ep 左右 loss_c 就都是 0 了，但實際看圖片，其實給得文字和生成出來的圖片仍然有不小的落差，而且圖片中出現的顏色有不小的變動，但 loss_c 還是都一樣是零，所以就改成用如前面描述的 classifier model，並把兩者結果用 softmax_cross_entropy_with_logits 的方式去計算 loss_c，這樣讓 classifier 學習效果似乎更好。此外，實際跑的時候也發現，discriminator 學習速度比 generator 快很多，所以到後來 loss_d 可以小到小數點後四位，然後 generator 就 train 不太起來了，此時 loss_g 的變動很大，也沒有穩定變小的趨勢，因此改用 wgan，把計算 loss 的方式改成如前面描述的 objective function，讓訓練更穩定。

- Experiment settings and observation (2%)

左圖是原本 model 訓練到 600ep 的結果，儘管已經訓練了兩天，但還是圖片線條簡單，也和給的文字不太符合，尤其第五六行部分，文字是藍頭髮和紅眼睛，但訓練出來的 16 張圖卻大多是偏紫頭髮紫眼睛，而右下圖是使用改進後的 model，訓練了一天，就真的有出現藍頭髮和紅眼睛，雖然頭髮還是偏藍綠色，但已經比起原本的 model 效果更好，也學得更快。

- Style-transfer(2%)

Style-transfer 的部分，我是從原本 faces 的資料集中隨機找出 30 張 label 有藍色頭髮的圖片作為要轉換的圖片，並把剩下其他顏色頭髮的圖片轉為有藍色頭髮，以下是跑出來的結果，左邊是原圖，右邊是 Style-transfer 後的結果。