

ADLxMLDS 2017 hw3

工科所碩三 葉峻孝 r04525061

Model description (2%)

Must include model structure, objective function for G and D

我把實作架構分為: Data_Preprocessing, Model_Structure, Model Parameters

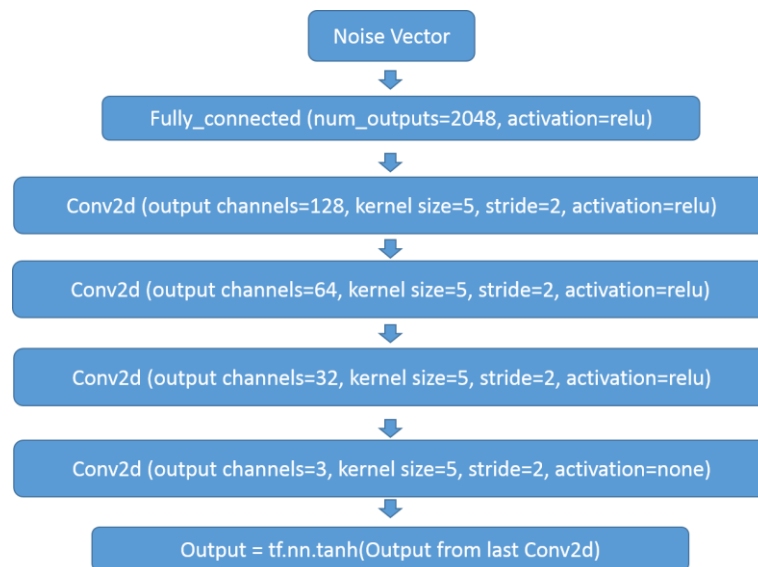
Data_Preprocessing

我主要是先過濾取出有包含<hair>, <eyes> tag 的圖來做處理，其他沒有以上頭髮或眼睛 tag 的圖我就省略不運用。我有運用助教在 PPT 上面所給出的頭髮與眼睛的顏色，我就只把這些取出來用，其他沒有碰到的顏色或是只有<hair>或<eyes>的圖片我也不取用。剩下符合資格的圖片大概有 5000 張。然後我針對每個顏色給他一個值去建立 one-hot encoding。例如輸入是 blue hair, green eyes，我 encode 過對這些 tag 去標註 label，就會變成 8, 2(hair), 20, 3(eyes)。所以我把每張圖片都 encode 過後就送進 model 作訓練。

Model_Structure

我把 model 分為 Generator 與 Discriminator 來做討論。

Generator:



我主要 Generator 的架構為一個全連接層接上 4 個 Conv2D 層。輸入為一個雜訊向量，進去一個通道為 2048 的全連接層，再來接上 4 個 Conv2D 層，輸出的 channel 個別為 128, 64, 32, 3，然後 activation function 都是使用 relu。最後接上一個 tanh 得出 Generator 的 output。

我的 objective function 為:

$$\min E_{h \sim p_h, z \sim p_z(z)} [-\log(D(G(z, h)))]$$

主要是取 E 的最小值，E 當中所包的就是先取 D 當中的 G 的 output，G 的 input 裡面有雜訊 z 還有 tags。

Discriminator:



主要 Discriminator 的架構為 5 個 Conv2D 層。輸入為訓練圖片，進去接上 3 個 Conv2D 層，輸出的 channel 個別為 32, 64, 128，然後 activation function 都是使用 relu。這步驟做完後，我把輸出的圖片特徵向量結合 tag 的向量，再來接上兩層 Conv2D 層，輸出通道分別為 128, 1。得出 Discriminator 的 output。

我的 objective function 為:

$$\begin{aligned} \min - \{ & E_{x, h \sim p_{data}(x, h)} [\log D(x, h)] + E_{x \sim p_{data}(x, h), \hat{h} \sim p_h, h \neq \hat{h}} [\log(1 - D(x, \hat{h}))] \\ & + E_{h \sim p_{data}(x, h), \hat{x} \sim p_x, x \neq \hat{x}} [\log(1 - D(\hat{x}, h))] + E_{h \sim p_h, z \sim p_z(z)} [\log(1 - D(G(z, h)))] \} \end{aligned}$$

主要能看到有 hat 的就是生成(fake)的圖，沒有 hat 就是原圖(real)。X 是圖片，h 是所代表的 tag 文字。

Model_Parameter:

batch_size = 128

AdamOptimizer with learning rate = 2e-4, beta1 = 0.5, beta2=0.9, momentum=0.5

z_dim (noise dimension)= 100

epoch = 75

How do you improve your performance (2%)

1. 我先針對 `data_preprocessing` 這個部份去做改進，因為我發現有些圖像會標有兩種以上的頭髮或眼睛顏色的 `tag`，在這個部份我直接先把有符合的圖像去除，因為這樣會影響電腦去判斷此圖的 `label`。
2. 再來為了更加提升生成圖片的精準度，我把只有標註頭髮或只有標註眼睛顏色 `tag` 的圖片也去除，因為如果人工再去標註必定會花費許多時間。所以在前處理我都先加以去除。
3. 我在前處理有實作把動漫圖像的背景去除變成黑色的，因為我想說如果這些背景要去生成圖片，必定會對生成出來的圖片有一定程度的干擾或影響。我在一開始做實驗時，生成出來的圖片通常都會有些根本不是頭像該出現的 `pattern`。但我在實作去除背景時，發現其實運用切除邊框區域的方法最實用，因為我觀察到圖片頭像大多都會在正中間，我挑比例把邊緣做去除 (`padding` 補 0)，這樣就能消除一些背景資訊，讓最後生成圖片的雜訊更加降低。
4. 我在實作完最基本的 DCGAN 後，嘗試作了 Least Squares GAN (LSGAN)，在 LSGAN 裡面，主要需要改動的就是 `objective function` 改成 `least square` 的形式。目標函數將是一個平方誤差，考慮到 D 網絡的目標是分辨兩類，如果給生成樣本和真實樣本分別編碼為 `a, b`，那麼採用平方誤差作為目標函數，對於 Generator 的 `loss function` 來說。它的 `loss function` 為：

$$\min_G L(G) = \mathbb{E}_{z \sim p_z} (D(G(z)) - c)^2$$

C 代表生成器要去欺騙 D 的數據

對於 discriminator 來說，`loss function` 為：

$$\min_D L(D) = \mathbb{E}_{x \sim p_x} (D(x) - b)^2 + \mathbb{E}_{z \sim p_z} (D(G(z)) - a)^2$$

B 代表其真實的圖片數據，a 代表其偽造的圖片數據。此函式利用平方誤差來找到最小的 D 值，能夠讓 `loss` 達到最小。因為之所以對其做平方誤差，可以讓 $(D(x)-b)$ 和 $(D(G(z))-a)$ 這兩個值放大，能夠更快的以訓練去收斂。以下會有 DCGAN 與 LSGAN 的實驗比較結果。

5. 我還有實作 WGAN，主要是能夠改善 GAN 比較不收斂的問題。主要是要要求 `f` 導數的絕對值先不要超過 K 的前提下，對所有能夠滿足 K 條件的 `f` 取到上界。然後再除以 K。

$$W(P_r, P_g) = \frac{1}{K} \sup_{\|f\|_L \leq K} E_{x \sim P_r} [f(x)] - E_{x \sim P_g} [f(x)]$$

Generator loss:

$$-E_{x \sim P_r}[f(x)]$$





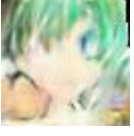
Discriminator loss:

$$E_{x \sim P_g}[f(x)] - E_{x \sim P_r}[f(x)]$$

所以 WGAN 對原始的 GAN 主要改進了四個要點。第一為在 Discriminator 的最後一層捨去掉 sigmoid。然後在 D 和 G 的損失函數不要對其取 log。然後每次更新 D 的參數之後，把參數的絕對值設定不超過一個閾值 c。最後捨棄掉 Adam 改換成 RMSProp。

Experiment settings and observation (2%)

接下來是我實作 DCGAN、LSGAN、WGAN 三種 GAN 的比較結果。

	Epoch=5	Epoch=50	Epoch=100	Epoch=200
DCGAN	 (blue hair, green eyes)	 (blue hair, green eyes)	 (blue hair, green eyes)	 (blue hair, green eyes)
LSGAN	 (blue hair, blue eyes)	 (blue hair, blue eyes)	 (blue hair, blue eyes)	 (blue hair, blue eyes)
WGAN	 (green hair, green eyes)	 (green hair, green eyes)	 (green hair, green eyes)	 (green hair, green eyes)

我運用了四種不同的 Epoch 環境來比較，這樣比較容易可以看到訓練的漸進性。從圖中可以看到在 Epoch 等於 5 時，因為 model 才開始訓練，所以三個 GAN 所產生出來的圖片都算非常的模糊與不太能表現出這是一張人臉。但是基本上所給的 tags 都大概能表現出來。例如藍色的頭髮與綠色的頭髮。在 Epoch 等於 50 時。比較能夠看出產生出來的圖片有人臉的樣子。然後我所下的 tags 大多都能表現的出來。例如藍色的眼睛與綠色的眼睛。但在這邊就能看出 WGAN 在產生圖片時會有比較多的扭曲。而原先實作的 DCGAN 表現的就比較

好，而且圖片都比較清晰，不會有太多的雜訊。但訓練到 Epoch=200 時，反而我觀察到 LSGAN 都有比較多的雜訊，而且圖片較不清晰。我推測是訓練比較不穩定所致。

Bonus (2%)

Style Transfer:

我在這個部分實作的上面是使用這篇論文來當作我的方向: **Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." arXiv preprint arXiv:1508.06576 (2015).**

該論文使用兩個網路，一個是生成網路，另一個是描述網路，兩個網路我把他命名跟 GAN 一樣都叫做 G 和 D。主要是先提取訓練圖片的特徵，然後加入 z 雜訊，從當中得到 G 網路的特徵。提取特徵前要先計算裡面的 Gram 矩陣。

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l.$$

再來去計算所有的 loss function。

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l - A_{ij}^l)^2$$

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^L w_l E_l$$

這樣一個一個迭代去訓練就能夠下降 loss 然後利用 G 產生 style transfer 的圖片。

我實作上面主要是把助教給的 dataset 分成兩種比較不一樣風格的動漫圖像。以這樣去做訓練。我的目的是要從一種風格的圖片轉換成另外一種風格的圖片。轉換前:



轉換後:



這樣是利用臉型不同(都是 blue hair, green eyes)的轉換，雖然效果不是說非常好，但還是可以看出，有轉換的一點效果。