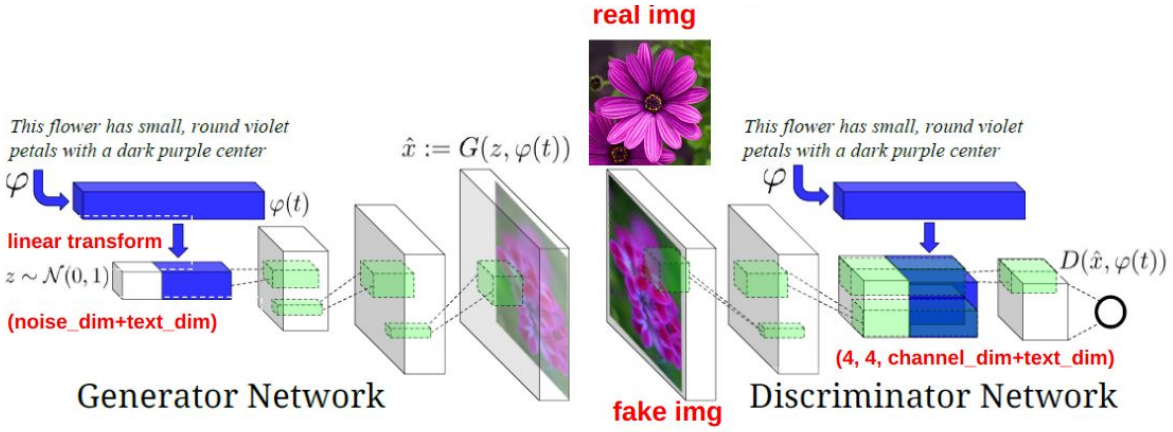
Model Description (2%)

我做的Conditional GAN 是使用DCGAN，基本架構跟助教在投影片裡面提供的一樣。input vector是透過glove來做word2vec。



Objective Function：

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))].$$

參數：

z_dimension:100,
caption_vector_sim: 600,
first convolution layer output: 64
first deconvolution layer output: 64
fully connected layer output: 64
Optimizer: Adam(momentum=0.7, lr=0.0002)

參考github: https://github.com/carpedm20/DCGAN-tensorflow

Improvements (2%)

1. **Input Vector：**
   在嘗試過後，我使用了glove的word2vec來進行input text的處理。

我最後選擇使用300維的glove pre-train model。讀進每一個caption之後，我截出頭髮以及眼睛的顏色部份，各查詢成300維的vector，在利用hstack，接成一個600維的vector當成input vector。

2. **Label Smoothing：**
   利用老師上課有提到的技巧，用這招可以train的比較好。將原本的Label(1,0)換成(0.9,0.1) 。這樣可以避免Generator一開始generate出來的data跟真實的distribution差太多。

3. **Update Frequency的選擇**
   看到許多別人的github都把Generator Update Frequency > Discriminator update Frequency 當作一個training tip。除了助教建議的2:1之外，我也嘗試了不同的update比例，並且對Loss作圖（詳見Experiments）。最終選擇仍為表現最佳的2:1。

4. **WGAN 嘗試：**
   我有嘗試使用WGAN的Loss Function 如下：

```
if self.options['model_type'] == 'wgan': # WGAN Model
    d_loss1 = tf.reduce_mean(disc_real_image_logits) - tf.ones_like(disc_real_image)
    d_loss2 = tf.reduce_mean(disc_wrong_image_logits) - tf.zeros_like(disc_wrong_image)
    d_loss3 = tf.reduce_mean(disc_fake_image_logits) - tf.zeros_like(disc_fake_image)
else:
    d_loss1 = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(logits=disc_real_image_logits, labels=tf.ones_like(disc
    d_loss2 = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(logits=disc_wrong_image_logits, labels=tf.zeros_like(di
    d_loss3 = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(logits=disc_fake_image_logits, labels=tf.zeros_like(dis
```

   並且試著配合WGAN的介紹Clip Discriminator的weight、把Discriminator Update Frequency調的更高。但是最後還是train不起來，結果非常的炸裂。

5. **Discriminator Loss**
   我有在Model中使用助教所提供的提示,也就是將 fake image- right text、real image- wrong text、以及 wrong image- right text 都當作錯誤的結果餵給 discriminator (其中還有Label Smoothing，下一點中將會提到)。我的做法是，最後在計算loss的時候，把三個 loss 相加起來成為一個 loss。

```
d_loss = d_loss1 + d_loss2 + d_loss3
```

   (d_loss1, d_loss2, d_loss3 即為上一段code中的三個loss)。相加後當作新的discriminator Loss 來做優化。

-Experiments (2%)

- **Eyes Color vs Hair Color**

  我的DCGAN最後train起來有一點Model Collapse的情況發生，即同樣的text之下會產生幾乎一樣的output。我也藉此機會比較這個model同樣髮色下，不同眼睛顏色產生的圖片變異度（以及相反），藉此看出哪些顏色對model會有比較大的影響；又有哪些顏色是model有成功學習到「顏色」差異：

  Same Hair(pink):

  green **blue** **red** **yellow** **brown**

  可以發現綠色以及紅系色比較成功的讓model學會了變更「眼睛」部位的顏色，這些顏色的vector在我的model中比較成功的被辨識成「顏色」。反觀「藍色」的vector就讓我的model整個變了，被解讀為不只是顏色這樣的資訊。

  Same Eyes(green):

  grey **white** **black** **blue** **blonde**

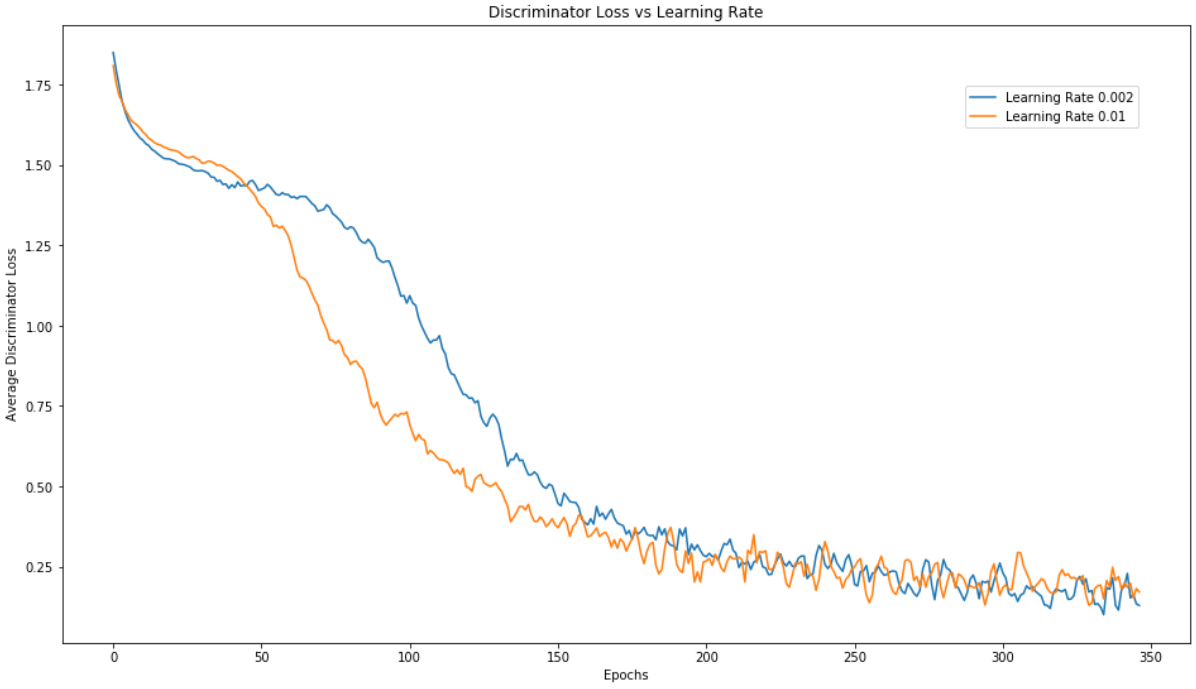
  

  可以發現，頭髮的vector資訊在model中被當作改變整個人臉比較重要的features，可以能事因為頭髮面積較大，所以在訓練的時候model就將這些vector中一部分資訊解讀為整張圖的架構。

第二部份我一共針對我最基本的GAN model做了三個參數調整的實驗並對Discriminator Loss作圖如下(用Discriminator的loss比較看得出下降的趨勢。)。

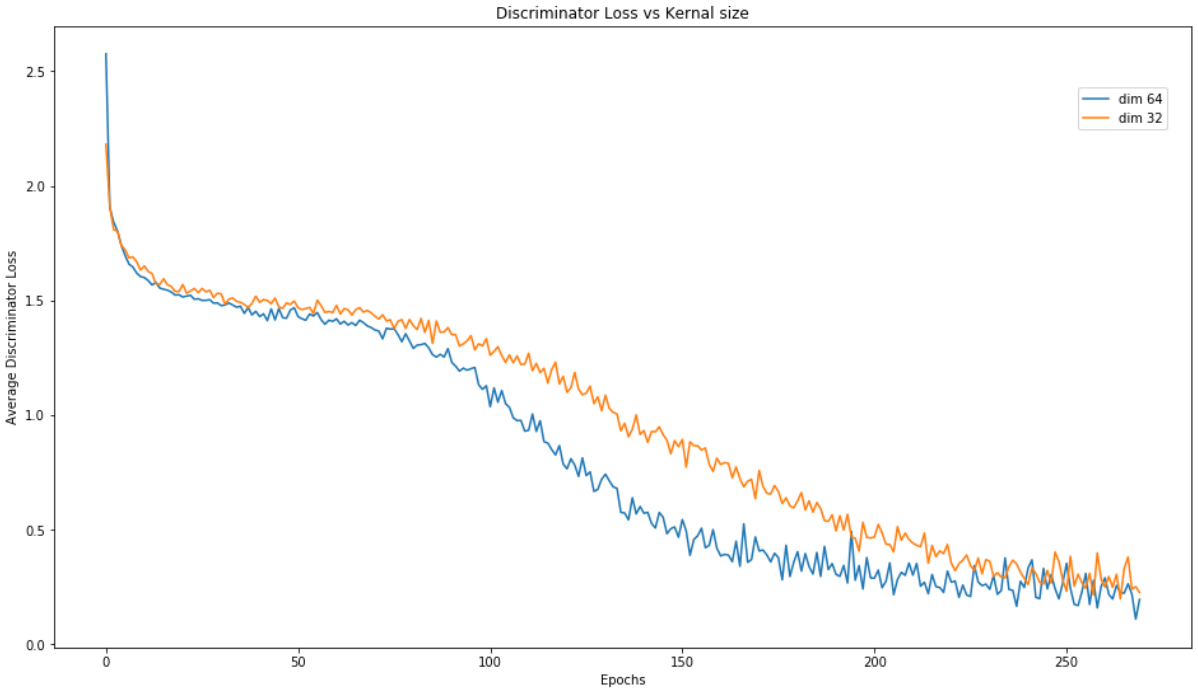
- **Learning Rate**

  在DCGAN上，Adam Optimizer中調整不同的Learning Rate。可以看出learning rate 使用0.0002的時候，會在300個epoch之後有比較好的進步程度，因此在我最後的model是選擇比較小的learning rate來做training。



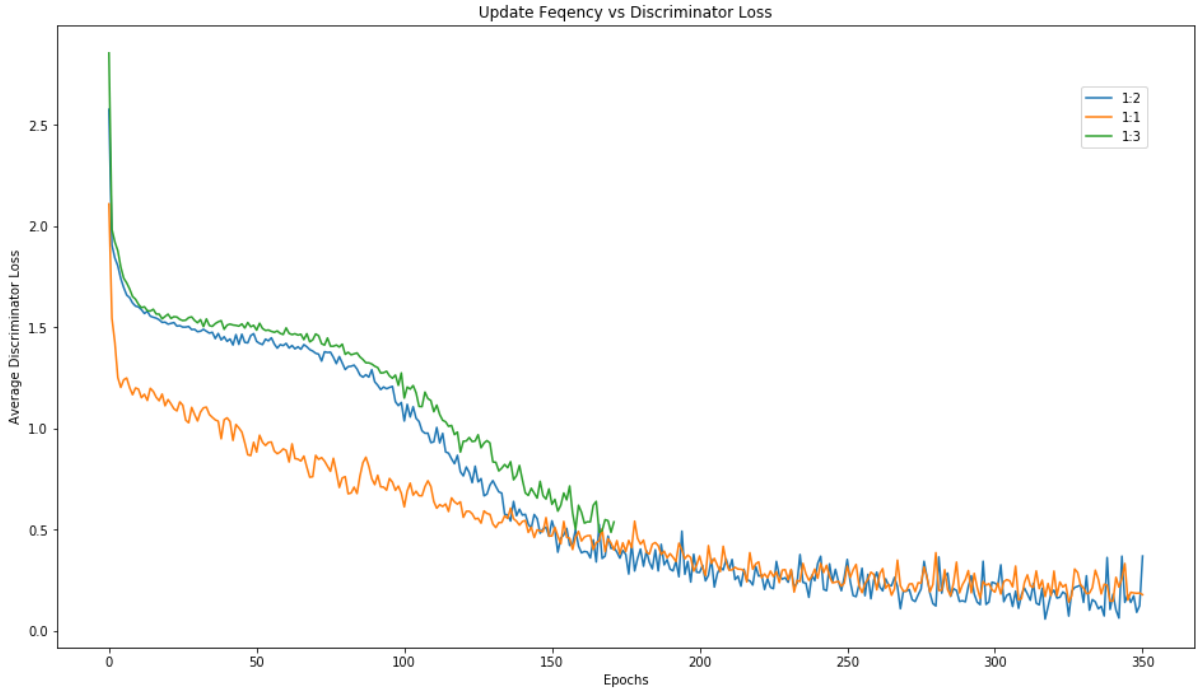
- **Kernel Size**

  實驗了不同大小的CNN kernel size對於training Loss的影響。發現開64 dimension的時候還是可以獲得比較好的訓練成果。



- **Update Frequency**

  最後是關於所謂Discriminator Update Frequency vs Generator Update Frequency 的「經驗法則」驗證。按照原始Paper裡面的資訊，是建議我們使用1:2這樣的比例。於是我自己試著做了1:1以及1:3的training並比較他們Discriminator Loss隨epoch關係繪圖如下：



  可以發現，1:2的這個比值最終真的有比較好的訓練效果。使用1:1訓練時，儘管前面可以下降很快，但是道地150個epoch之後就已經輸給1:2的model，到了300個epoch之後，training loss可能差到0.1。可見paper寫的還是乖乖遵守比較好，最終model也選擇1：2這樣的比例進行訓練。