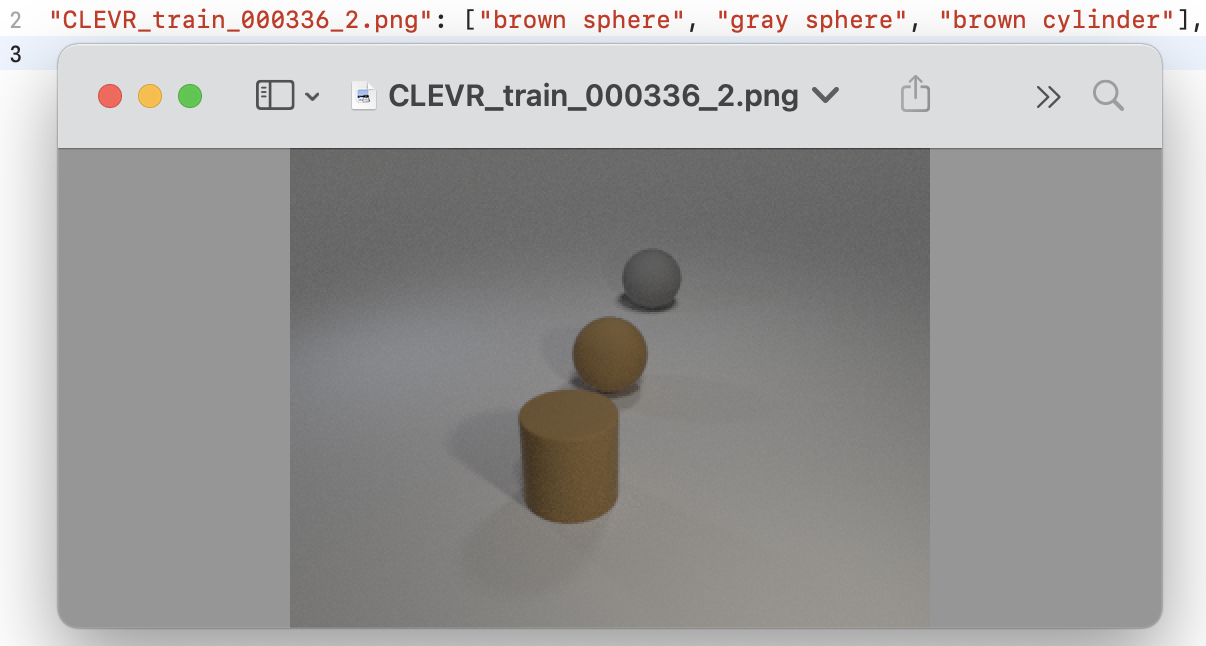
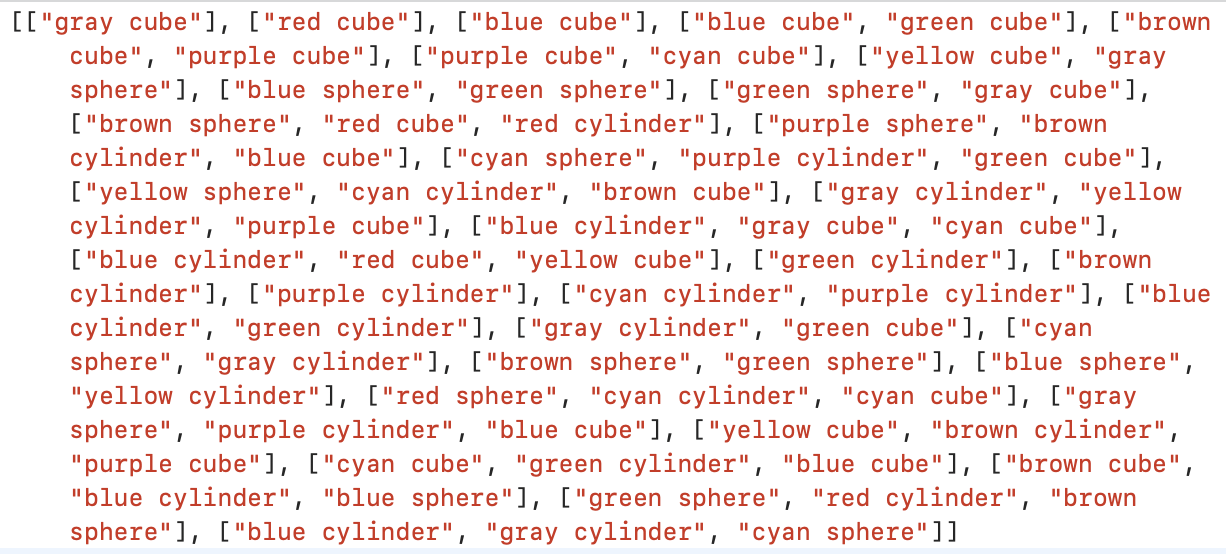
Introduction:

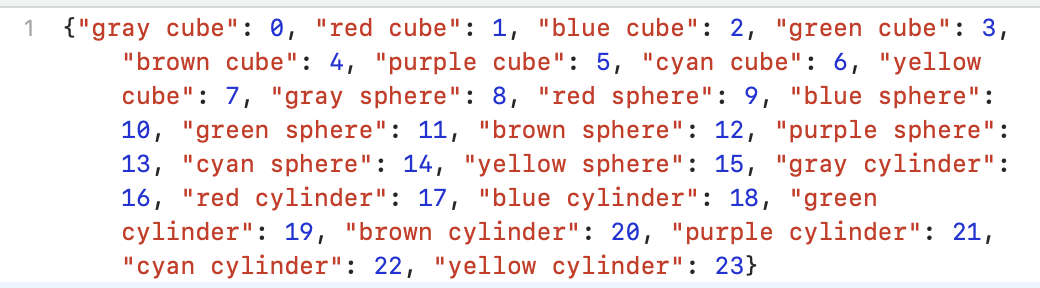
本次作業是要實作一個conditional GAN產生出所需要的圖片，condition labels的種類共有24個，一種物體對應著一種label，且有時可能會同時出現多個condition在同一個圖片當中，也就是說model要有能夠處理multi-labels的能力。

在訓練集中，種共有18009個images，每張image都有對應的label，測試集集是給予32個label用來當作condition作為給予model的輸入如圖一，其image與labels的對應關係如圖二。

圖一，圖片與labels的對應。

圖二，測試集所提供的labels。

為了方便使lables送入model中，我們還需要將condition轉換成長度為24的one hot vector，圖三則為用於轉換之table。



圖三，用於將condition labels轉換至one hot vector的table

再經由多次訓練之後，model便可以依照送進來的condition，做相應的圖片生成，如圖四為模型經由58次迭代的結果。



圖四，實驗生成圖片之一，於epoch 58

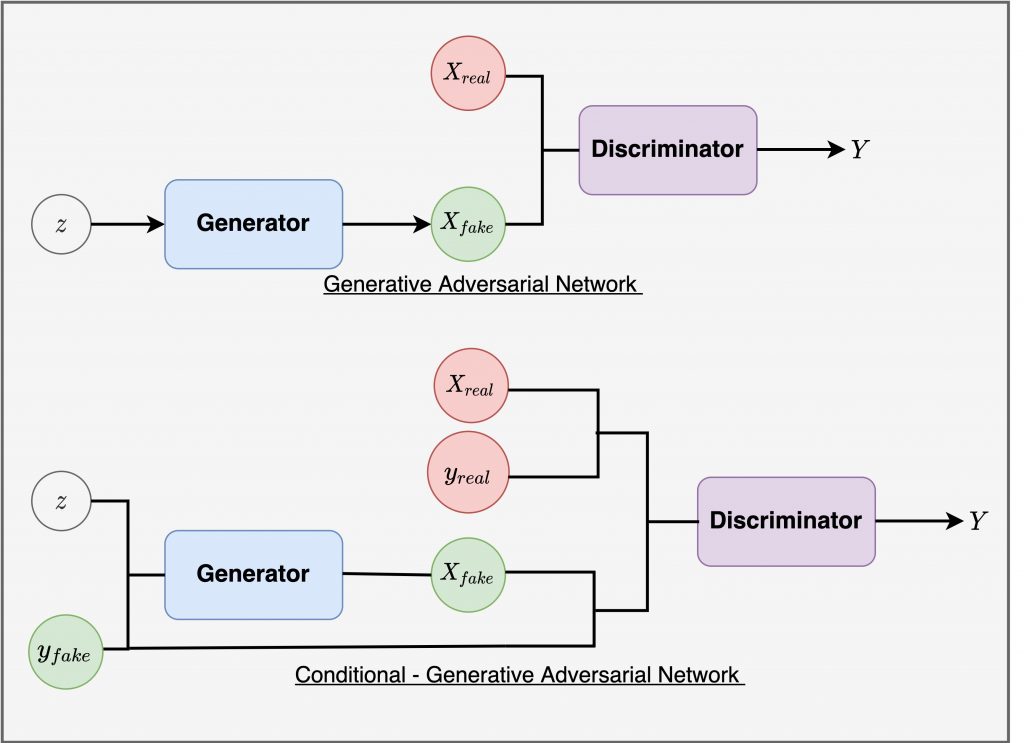
Implementation details:

這次我所與用的GAN模型架構為WGAN-GP，WGAN大致上的模型架構繼承了DCGAN的卷積網路用於生成及鑑別，但有別DCGAN是一個well-trained的模型，亦即是貢獻了一個很好訓練的模型架構及參數，WGAN則是從源頭上解決了origin GAN難以訓練的問題。由原生Origin測量模型本身的分佈與訓練資料的分佈的JS divergence，改為EM distance，前者無法提供分佈之間的相應距離，也就是說沒有辦法判斷，模型提供的分佈跟訓練資料的差別。後者則可以給予，給予了模型能夠測量的依據。WGAN-GP則是在WGAN的基礎上進行了改良，加上了gradient penalty來取代WGAN的weight clipping，並在生成的圖片中間上了gaussian noise。

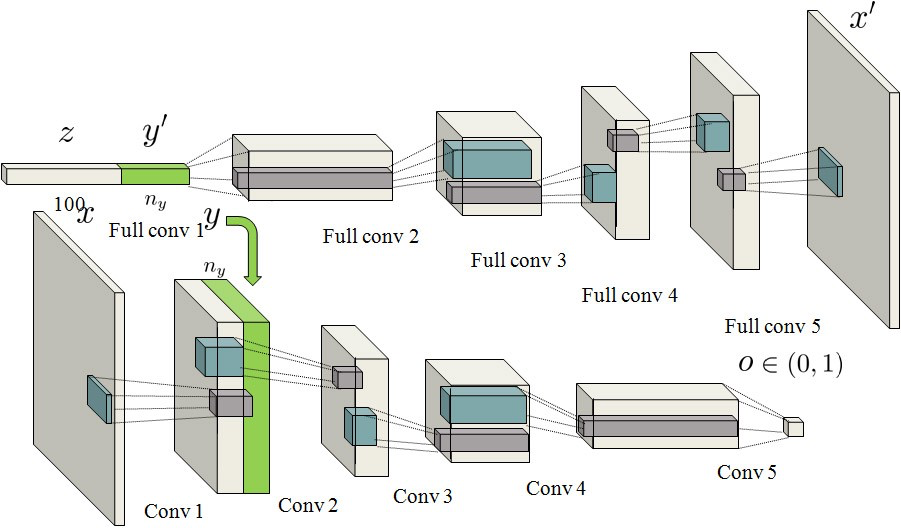
而如果要在上述所提到的GAN中加上condition的功能常見得實作方式有以下兩種：

一，可以將所有可能出現的condition送入embedding中，也就是說先計算所有可能出現的種類，例如有24個分類所有的可能就是2^24，但是所提供的訓練集中有提到，一次最多只會出現不超過四個的condition，整體的數量會更少，將每一個可能出現的condition做編號，送至image \* image的embedding中，在與原本image的圖片做相連，也就是說把原本3\*64\*64的image加上1\*64\*64的embedding condition 變成4\*64\*64再送入model中，便達成image + label的效果。

二，可以將原本condition labels的one hot vector直接送入一個24 to image\*image 的linear fc中再將這一維向量轉換成image \* image的二維張量，一樣如方法一加入成圖片的第四個維度。整體架構如圖五、圖六。

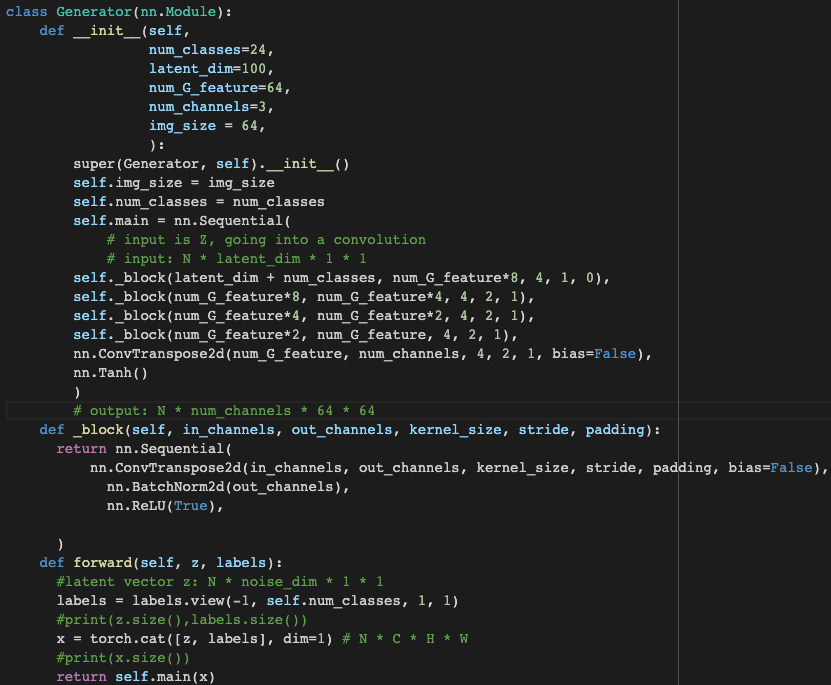


圖五，GAN與cGAN

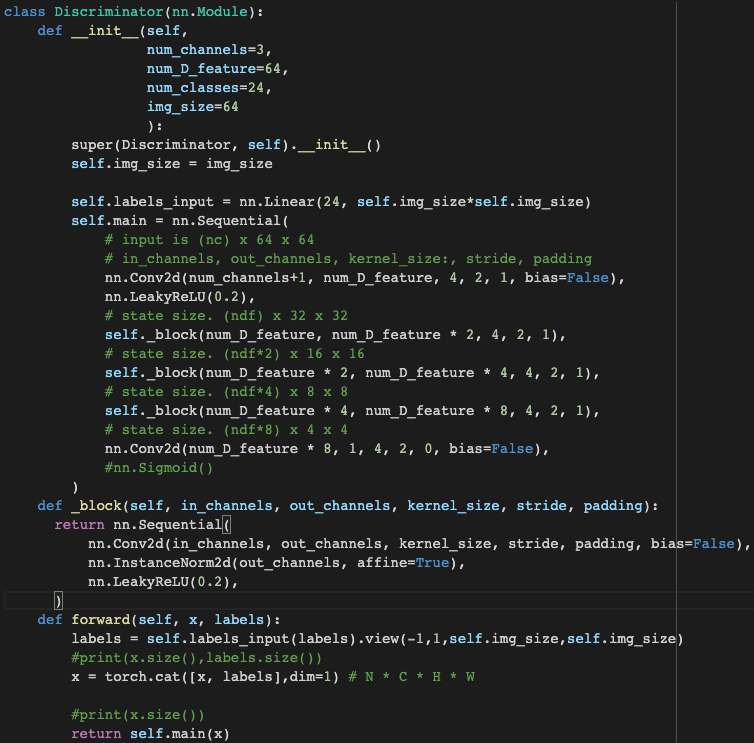


圖六，conditional DCGAN

而基本上我的Generator與discriminator基本上跟DCGAN的差距不大，如圖六、圖七。

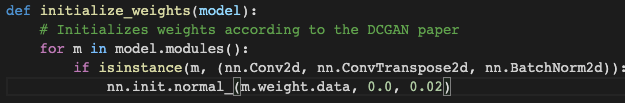


圖六，Generator架構。

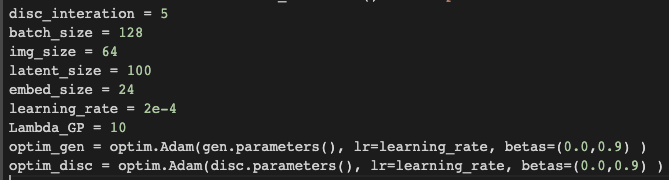


圖七，discriminator架構採用InstanceNorm並移除DCGAN中的sigmoid。

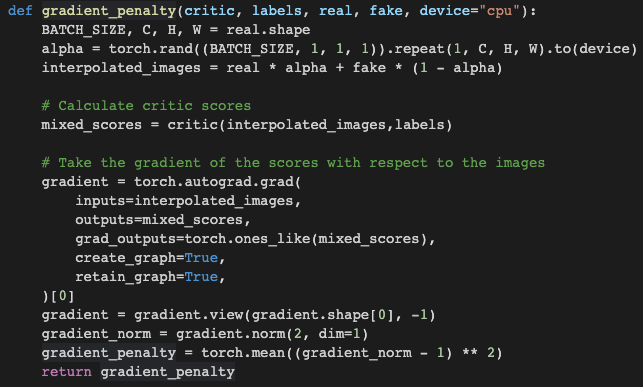
在訓練之前需初始化model的weight此參數由原始paper提供。如圖八。



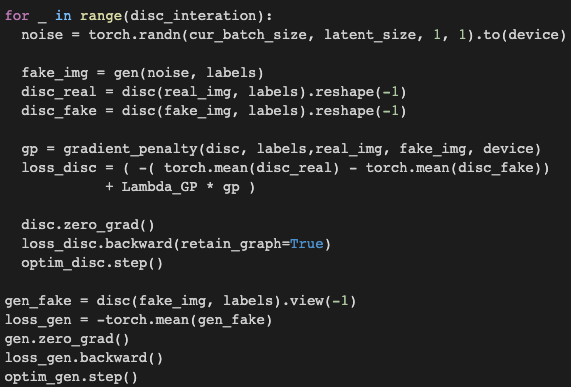
圖八，初始化model的參數



圖九，model所使用之hyperparameters，為論文提供。

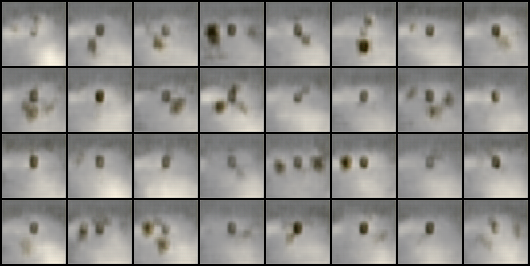


圖十，WGAN-GP中的gradient penalty。



圖十一，因為要在同一個epoch中把discriminator多訓練幾次，因此將loss=G\_loss+D\_loss拆開，並在D\_loss中加入GP作為訓練的一部分。

圖十二，epoch 0。



圖十三，epoch 4。



圖十三，epoch 8。



圖十四，epoch 24。



圖十五，epoch 40



圖十二，epoch 86、test\_acc=0.625

Discussion:

* 通常論文所提供的hyperparameter通常都是最好訓練的，修改參數絕大多數都不會比較好。
* 將實驗中，將discriminator的InstanceNorm改成常見的BatchNorm，前者在前中期的訓練效果中會緩慢的提高acc，且G\_loss、D\_loss穩定，但後者往往在中後期會發生梯度爆炸的情形，G\_loss、D\_loss會飆高。
* Generator與Discriminator的架構最好是完全對稱的，不同的卷積結構會導致latent space中gaussian noise的解讀不盡相同，導致結果不優。
* 實驗中，如果Generator和Discriminator的訓練次數如果是1:1的話會導致模型在前期一直產生不出具有意義的圖片，生成出的圖片就如noise一樣，相較訓練次數1:5會需要迭代很多次，其原因可能為，如果discriminator連最基本的判斷能力都沒有的會，generator的練訓會變得毫無方向。
* DCGAN非常吃原始paper所提供的參數，任何一點參數的改動或是是G和D的架構變化，對DCGAN來說都是致命的，輕則模型表現不好、重則模型整個無法訓練，DCGAN感覺更像是提供了一個經過爆搜所找到一組參數及架構，但也沒有一個依據作為訓練的標準。
* WGAN-GP相較WGAN、DCGAN訓練上較穩定不大會出現G\_loss、D\_loss爆炸的情形，且訓練出的acc看來都有穩定且緩慢的上升，至少都會跟先前一樣好。
* WGAN-GP因為有EM distance和gradient penalty的緣故，訓練上感覺會有一個穩定的方向可供參考。
* 在實驗過程中發現，通常單一labels的情況會先訓練好，再來依次為兩個label，以此類推上去。換句話說，模型感覺會先學會一個物體的圖片，再來為兩個物品，及更多。
* Colab pro真爛，老是不讓我連GPU。