人工智慧 作業四

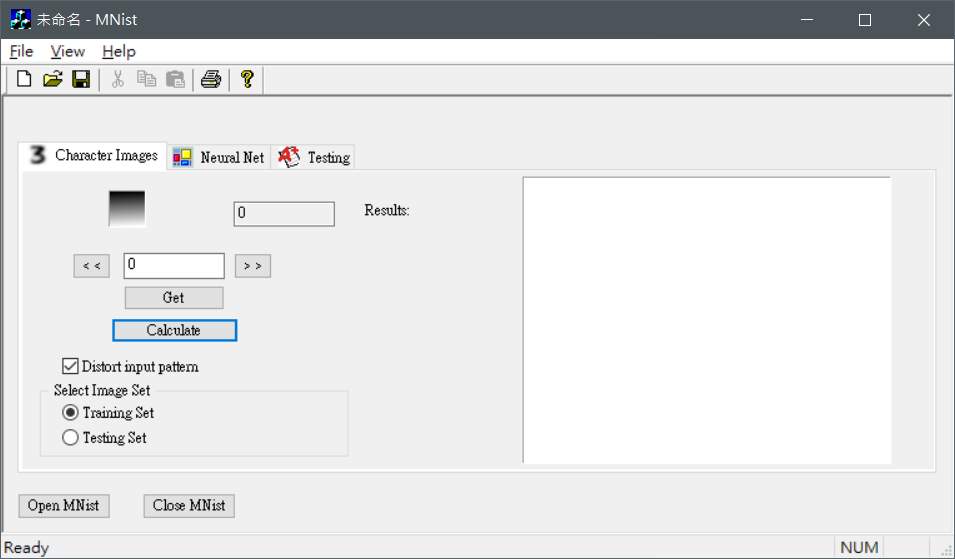
資工112 40847015S 紀軒宇

A. 首先，你要說明怎麼做才能開始這個專案？(看報告的人也不一定用過，所以要說明一下。) 注意：請說明如何執行這個程式，並詳細說明你所使用之機器軟硬體規格、 所用的作業系統等相關資訊以及你為何選擇這樣的規格。另外請提供你的連絡電話，以便不時之需。

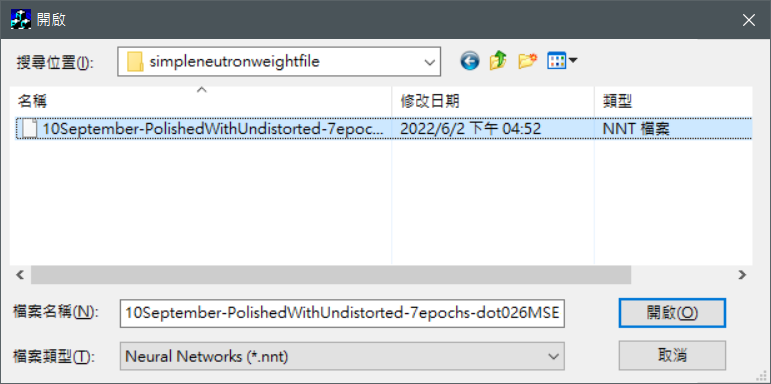
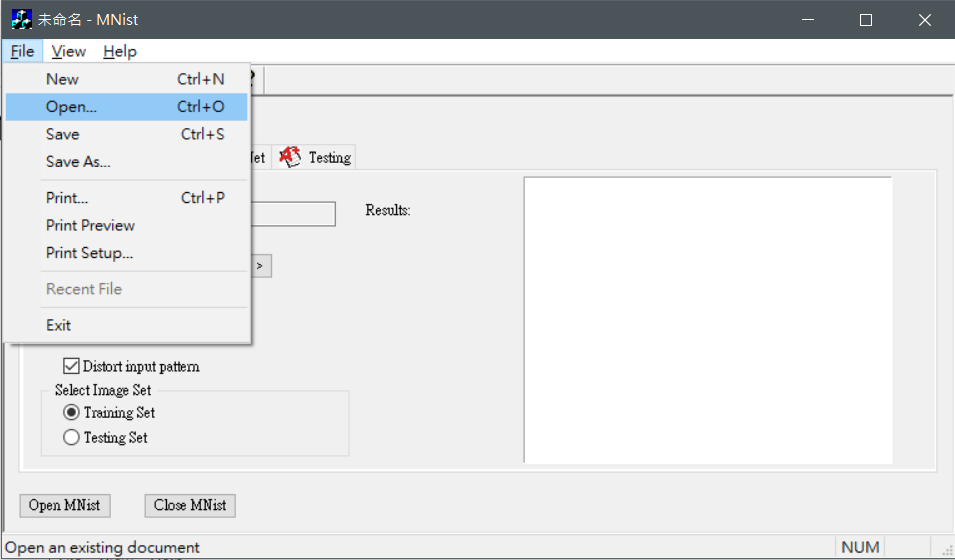
- 如何執行

首先，從https://www.codeproject.com/Articles/16650/Neural-Network-for-Recognition-of-Handwritten-Digi#Architecture下載主程式、範例權重及MNIST 資料集。

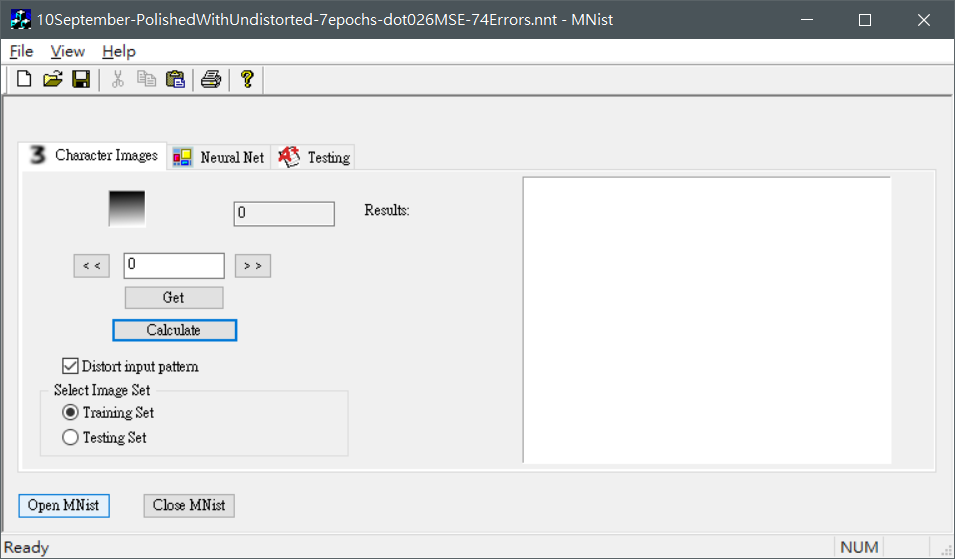
將所有東西都解壓縮後，接著點開主程式（存放路徑\Demo-MNist\Unicode Release\NMist.exe）可以看到程式介面如下。

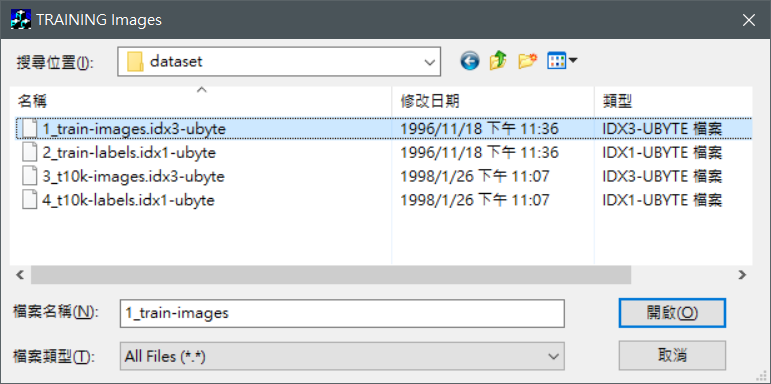


接著要打開權重的檔案，點開File>Open，選取作者提供的權重範例（.nnt檔），匯入後即可。



再來是匯入NMIST資料集，點擊左下方的Open MNist，接者他會要使用者依序選取訓練資料、有標籤（答案）的訓練資料、測試資料、有標籤的測試資料，這些檔案都輸入進去後，即可開始使用本程式。





▲依序選擇四個資料

- Character Images

在匯入資料集後即可使用此部分，可以在輸入框中輸入資料編號按下Get，或是用輸入框旁邊的<<或>>選取上一個或下一個資料，下方有幾個東西可以選取

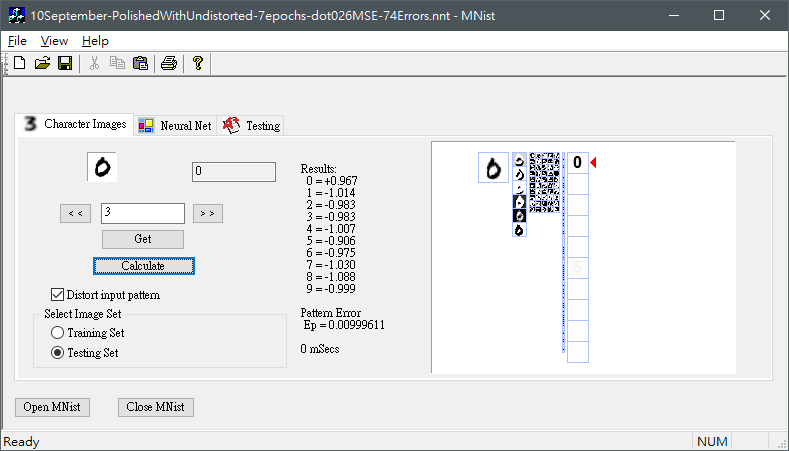
Distrort input pattern可以選取是否要對輸入資料做一些扭曲，使資料更接近我們平常寫字時可能會有的將數字寫的很扁、或是寫的斜斜的各種情況。

Select Image Set可以選擇剛剛輸入資料時的資料集是由Training Set中拿出來的或是Testing Set中拿出

接者是資料預覽部分，輸入框左上就會顯示剛剛選取的資料的圖像，右上方就會顯示此資料的標籤（答案）。

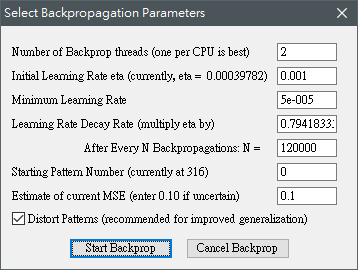
按下Calculate就會根據模型中的權重給出 Result，並顯示在程式中間，數值最高的即為越有可能的答案，下方也有Pattern Errror、計算時間，右邊的視覺化結果也會顯示整個NN的運作過程及最後輸出結果，並用紅色箭頭標示出最後認為的答案，從以下圖片中我們使用Testing Set的第3筆資料，並計算其結果，可以看出用作者提供的權重，出來的結果認為為0的可能最高，Result數值為+0.967，再來是6，數值為-0.906，最右邊的視覺化結果也可以清楚看到0的顏色最深、再來是5，依數值高低變化。

Pattern Error的計算方法如下所示，xi為Result中實際的10個輸出值，Ti為目標的輸出值，對於一個標示為0的資料，他在最後一層0號神經元的預期輸出即為1，其他預期輸出即為-1。



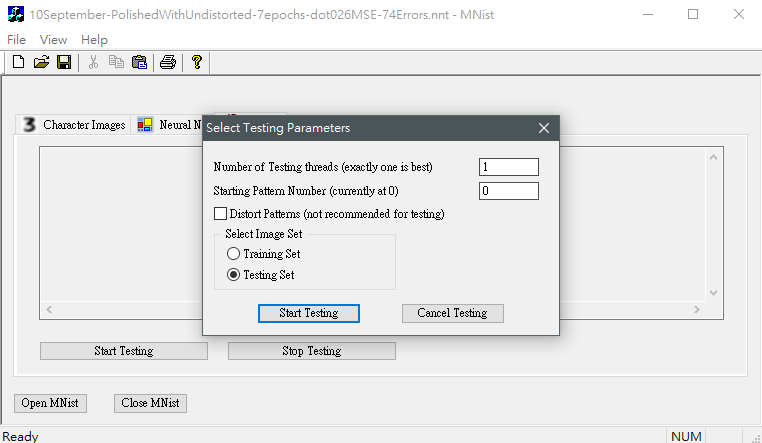
- Neural Net

訓練部分就是在這個區塊，在輸入好資料集後，打開（或創建）一個新的權重檔案，就可以開始訓練了，點擊Start Backpropagation，程式會給使用者許多參數設定，關於那些參數C部分會再做詳細介紹，訓練過程可以隨時停止並存檔，也可以將訓練到一半的權重檔案讀進來用新的參數繼續訓練。

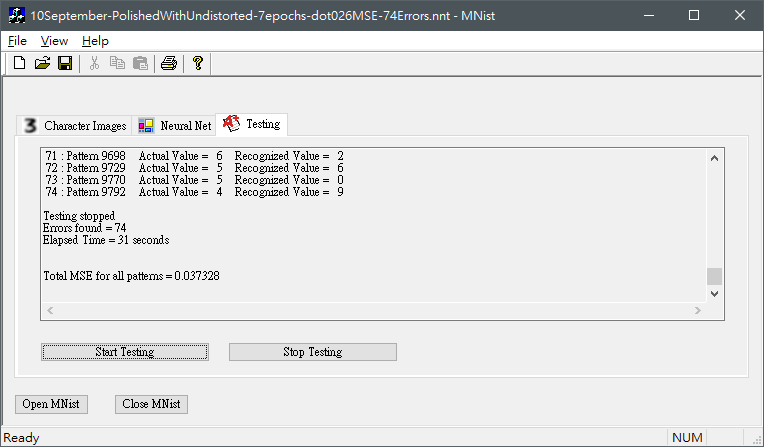


- Testing

Testing部分可以測試所有資料經過現有權重的輸出結果，並回傳所有輸出結果與標籤標示答案不同的資料、error等，也可以選擇要使用的線程數量，從哪一個pattern開始、是否要對pattern做Distort以及要使用的資料集，接著按下Start Testing即可開始測試。



測試完後，即可看到列出所有判斷錯誤的資料編號，找到的錯誤數量，測試時間，及這些資料的Error（這邊Error使用MSE）。



- 電腦規格

CPU: Intel® Core™ i7-8750H

GPU:Nvidia GeForce GTX 1060

ram:24g ddr4 2666

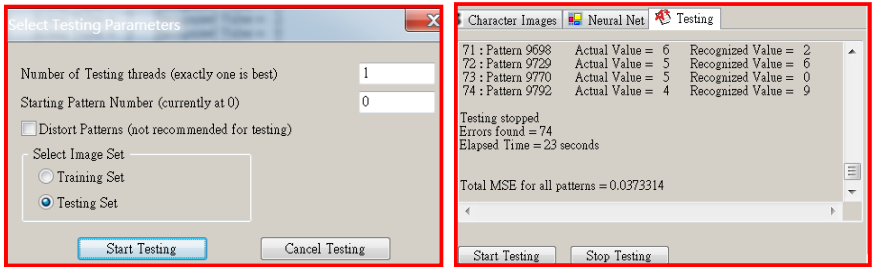
OS: Windows 10

開發軟體：Python 3.9.6

會選擇這樣的規格是因為當時考量未來可能會學習機器學習相關知識，選擇了有獨立顯示卡的電腦，方便日後學習。

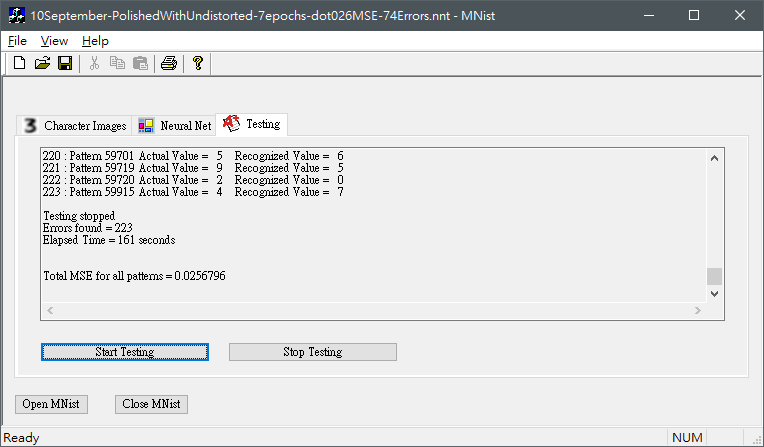
連絡電話：0988056412

B. 接著用O'Neill先生現成的權重表來測試，其結果到底真的如O'Neill說的那麼好嗎？ 你如何執行測試Training set及Testing set？測試花多少時間？正確率是多少？MSE是怎麼算的？MSE是多少？和O'Neill先生說的一樣嗎？其中Number of Testing threads 意思為何？能不能用大於1的值？若勾選Distort Patterns結果會變好還是變壞？

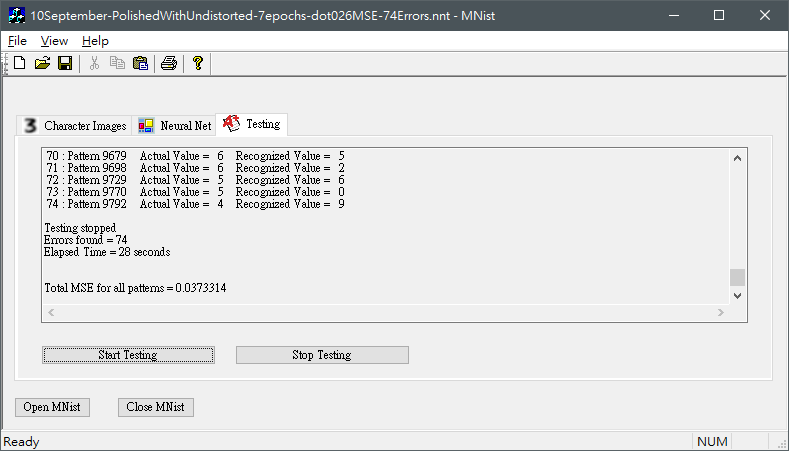


這邊使用A題中Testing的方法做Test，並分別對Train Set跟Test Set做測試，Distort Patterns的部分不勾選，MSE為均方誤差Mean Square Error的縮寫，這邊的計算方式即為A題Testing部分提到的Pattern Error (EP)，對測試的所有資料取其EP，計算其平均值。

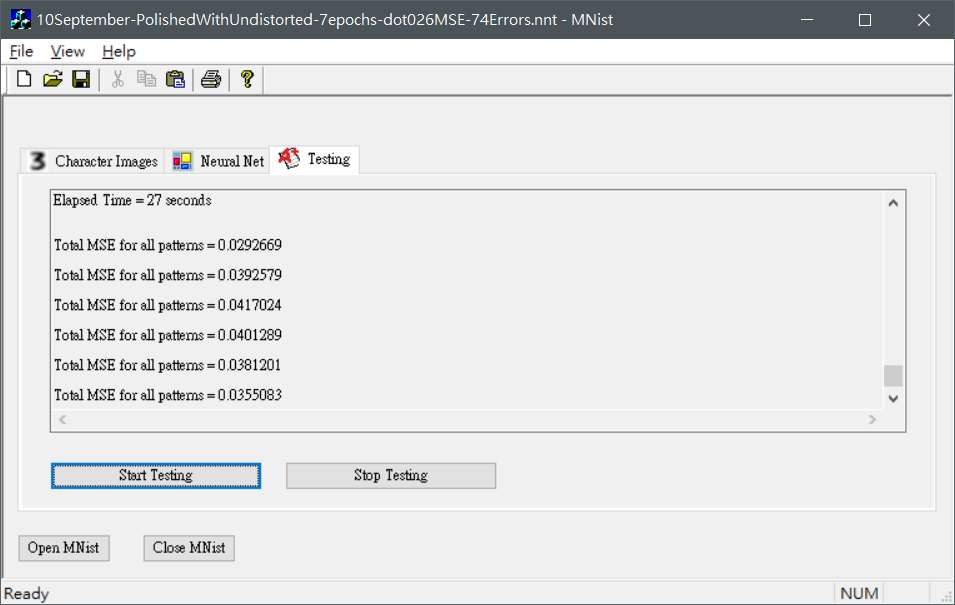
Train Set的測試結果如下圖所示，在Train Set的60000筆資料中，有223個錯誤，正確率計算為(1-223/60000) ≈99.63%，測試時間161秒，MSE計算約為0.026



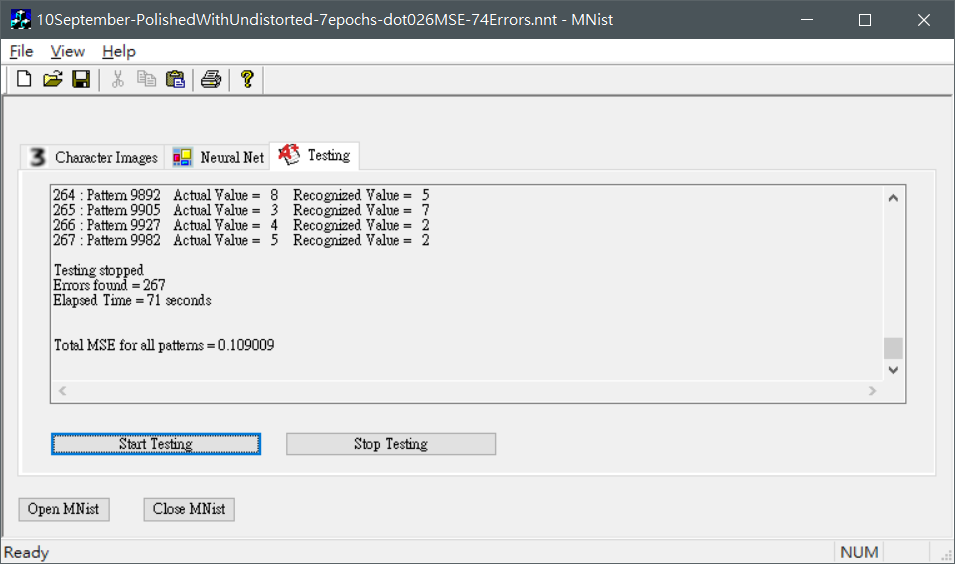
Test Set測試結果則如下，在Test Set的10000筆資料中，有74個有錯誤，正確率計算為 (1-74/10000)\*100%=99.26%，測試時間28秒，就如作者在網站上所寫的一樣。



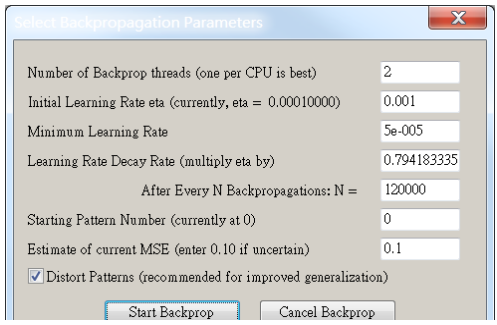
Number of Testing threads指使用多個線程進行Testing，這邊使用6線程對Testing Set做測試，結果如下圖，可以看出，即使使用了6個線程，但計算時間僅進步一秒，而且Total MSE也被分成六個。



勾選Distort Patterns，會使Pattern做變形，對Testing Set做測試，結果如下圖，可以看出錯誤率計算為(1-(267/10000))\*100%=97.33%，減少了2.03%



C. 接著你想自已訓練一下，說明你如何執行訓練及測試？如何得出正確率？你自已是怎麼算的？這個訓練結果花了多少時間？下圖中共有8行，接著請你依序詳細說明每行的專有術語的意義及用途為何？請你選擇性地改變其中一些參數，來做實驗。請說明能做到有進一步的進展、讓人驚嘆的結果嗎？



- 各參數意義

- Number of Backprop threads

計算時使用線程數量，在多核心系統上使用多線程處裡可以讓訓練速度更快。

- Initial Learning Rate

初始的Learning Rate，學習率過大會讓loss迅速上升，最後直接導致模型會發散或是最後模型只能收斂到一部分，如果過小，容易產生overfitting或收斂速度過慢，後面有一個參數可以在訓練過程中自動調整Learning Rate，後面會再說明。

- Minimum Learning Rate

最低Learning Rate，因為此專案中，在Learning Rate部分有加入Learning Rate Decay，所以Learning Rate會隨訓練時間增加而減少，此參數即為Learning Rate在Decay時控制，讓Learning Rate不會無止盡的減少。

- Learning Rate Decay Rate (multiply eta by)

剛剛提到Learning Rate Decay，Learning Rate Decay簡單來說，可以讓一開始使用的Initial Learning Rate較高，再隨著時間間少，可以在訓練一開始使用較大的Learning Rate加快前期的收斂速度，在訓練後期降低Learning Rate，使得模型能有更精準的收斂，這邊使用的方法是乘一個數字，使Learning Rate隨時間呈指數型下降。

- After Every N Backpropagations: N=

這個參數是在控制Learning Rate Decay多久要Decay一次。

- Starting Pattern Number (currently at 0)

這個參數是用來控制從Dataset哪一個pattern開始訓練

- Estimate of current MSE (enter 0.10 if uncertain)

一開始訓練的MSE

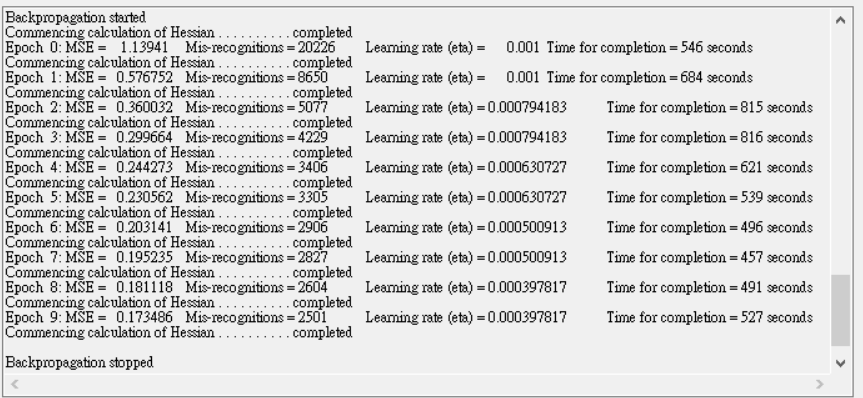
- Distort Patterns (recommended for improved generalization)

是否要對Dataset中的pattern做distort，如果勾選使用，可以讓訓練出來的模型更廣義化

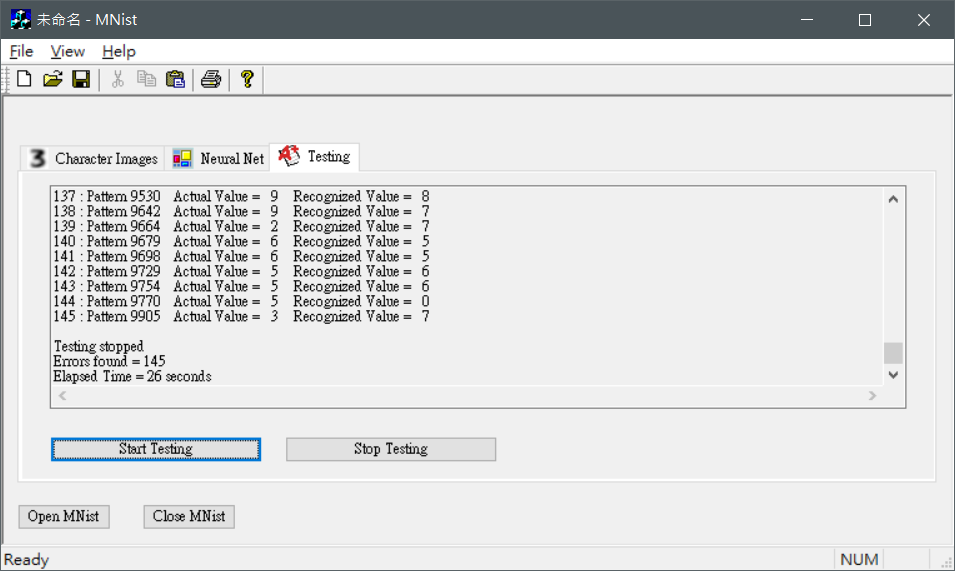
- 自己開始訓練

- 第一次訓練

第一次使用作者提供的預設參數，以供後面更改參數的比較，訓練10個epoch，訓練1小時39分鐘52秒。



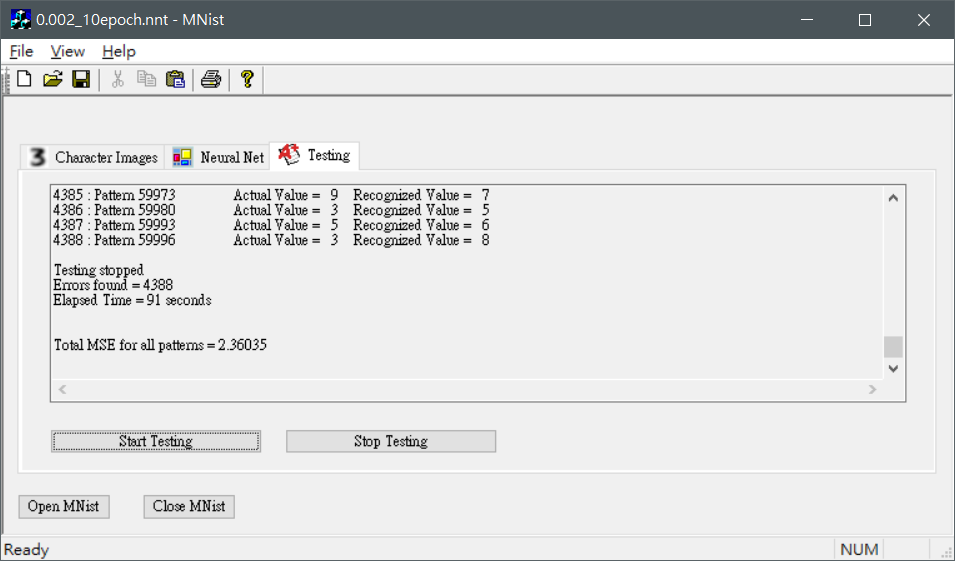
正確率部分計算為(1-(145/10000))×100%=98.55%，僅10個epoch就有良好的收斂表現。



- 第二次訓練

參數部分僅更改Initial Learning Rate，從初始的0.001改為0.002，訓練了10個epoch，跑了1小時45分鐘45秒，



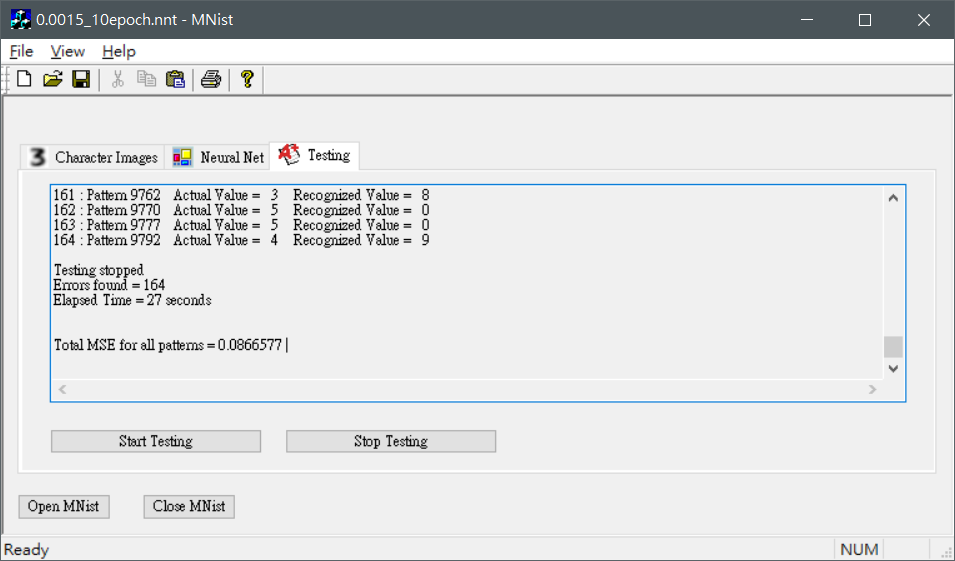


正確率部分僅有(1-(4388/10000))×100%=56.12%，雖然訓練期間的過程可以看到第3次epoch後MSE就有在逐漸下降，不過到了第10次epoch做完後的MSE還是比第一次epoch訓練時還要高。

- 第三次訓練

因為上一次實驗的結果發現將Learning Rate提升兩倍對於整個網路無法良好的收斂，所以保守一點，將初始Learning Rate改為0.0015（即預設值的1.5倍），一樣訓練10個epoch，訓練時間1小時29分鐘1秒

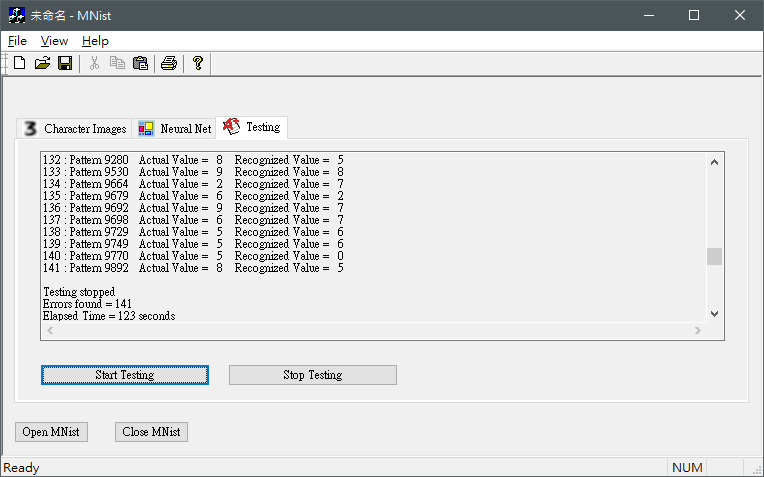
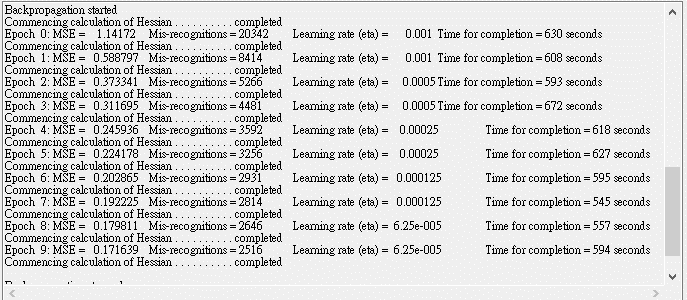




正確率部分計算為(1-(164/10000))×100%=98.36%，在相同的訓練epoch下已經有逼近使用初始參數時的正確率。

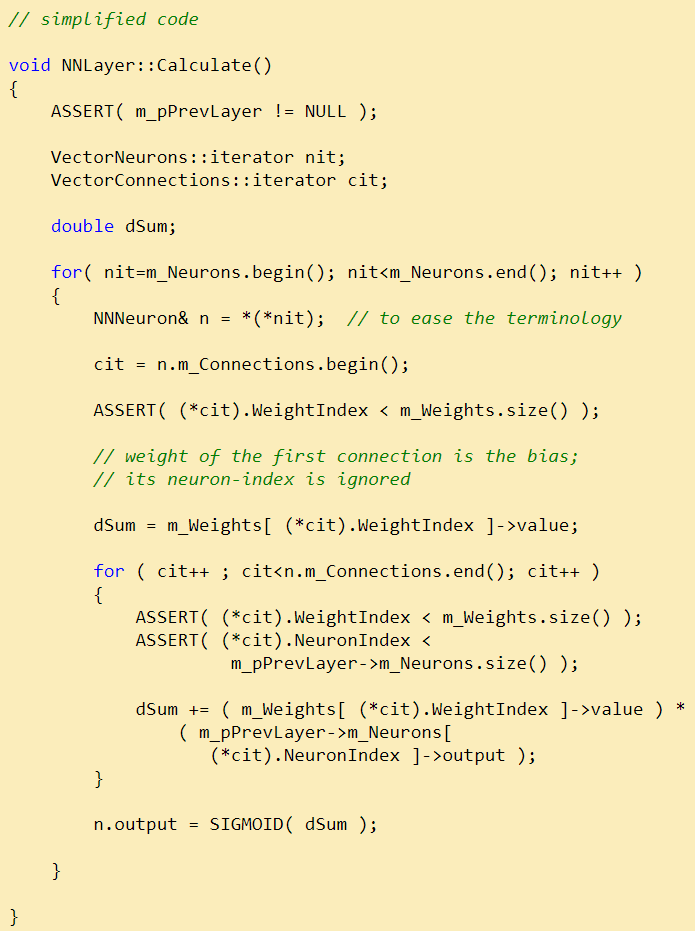
- 第四次實驗

前兩次實驗中可以看出Learning Rate因為設定較大，所以無法精準的收斂好，所以這次更改Learning Rate Decay，讓Learning Rate每120000次Backpropagations後減少一半，一樣訓練10個epoch，訓練時間1小時40分鐘39秒，可以看出由於一次減少的Learning Rate較多，所以整個模型有更精準的收斂。



正確率部分計算為(1-(141/10000))×100%=98.59%，在相同的訓練epoch下已經有比使用初始參數還要高的正確率。

D. O'Neill先生的網頁裡有8段程式碼，請你就你感興趣的、或有能力解讀的，任擇一段 中的一部份程式碼加以了解，說明那一部份在執行甚麼工作。

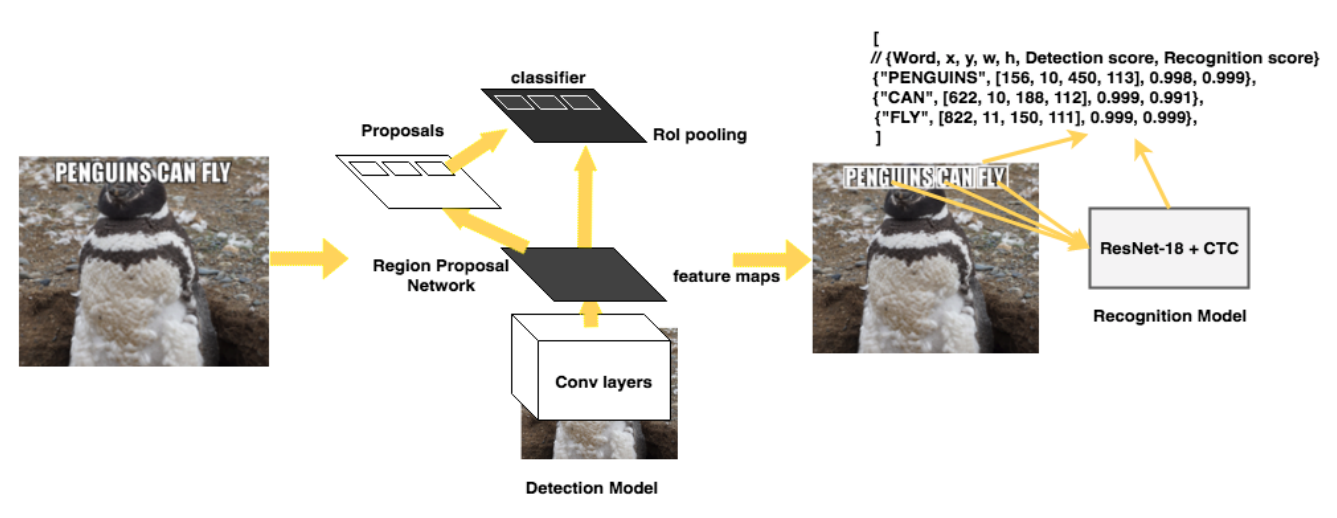


這一段主要是對神經元中所有的連接取得與之對應的前一的神經元權重以及輸出去計算該神經元的輸出。

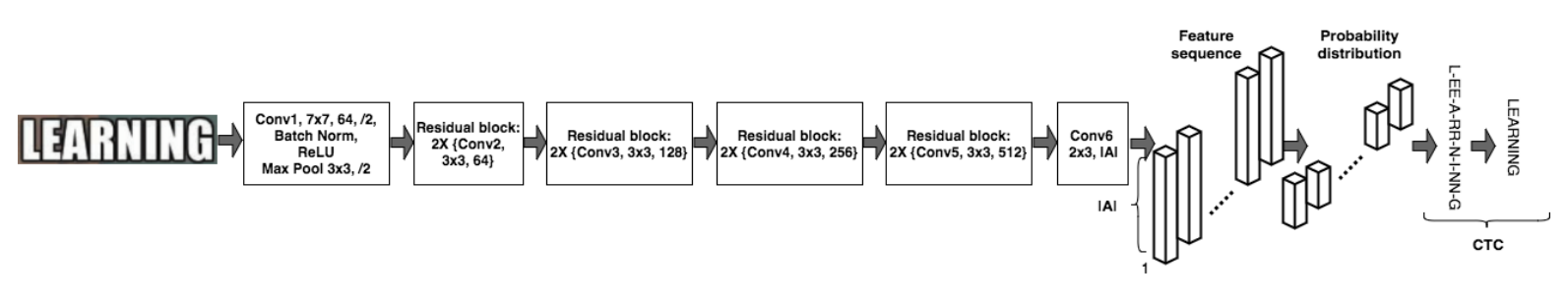
E. Convolutional Neural Network做為「手寫數字辨識」的用途，已經非常理想了。它也被推廣應用在辨識各國的文字辨識上。請在網站上蒐集這方面的最近的成果，做整理報告(最好是最近一年內的結果)。請盡可能列出以下資訊：它的CNN有多少層？神經元有多少個？有多少個權重?辨識的正確率為何？訓練多久？...等相關資訊。

除了一般的手寫文字辨識上，如果可以輸入一張含文字內容的圖片，可以直接將圖片中的文字提取出來，在應用上，可以自動將為網路圖片中上一些標籤，讓搜索引擎有機會直接透過想搜索圖片中的文字搜尋到該張圖片，也可以應用於社群軟體的垃圾訊息之過濾，讓將不實資訊透過內嵌在圖片中的資訊無所遁形，這邊要介紹的，就是由臉書所發表的大規模機器學習系統 Rosetta。

整個系統先將整個圖片用R-CNN辨識出圖片中有文字的位置，將那些位置的圖片碎片再輸入進另一個CNN辨識其文字內容，過程如下圖所示。



其中的文字辨識部分的神經網路架構如下，可以看到，因為不是單純的數字，更甚至不是單個文字，所以在網路的的部分遠比手寫數字辨識的複雜許多，對於模型的效能，採用準確性及Levenshtein 距離，來衡量辨識的結果。



結果如下圖所示，系統將所有有文字的地方成功框了起來，並成功的辨識出框框中的文字。



F. Convolutional Neural Network除了「手寫數字或文字辨識」的用途以外，還有其它的應用嗎？請你找出一個*其它的應用*，並說明這應用最近的結果(最好是最近一年內的結果)為何？它的CNN有多少層？神經元有多少個？有多少個權重?正確率為何？訓練多久？...等相關資訊。

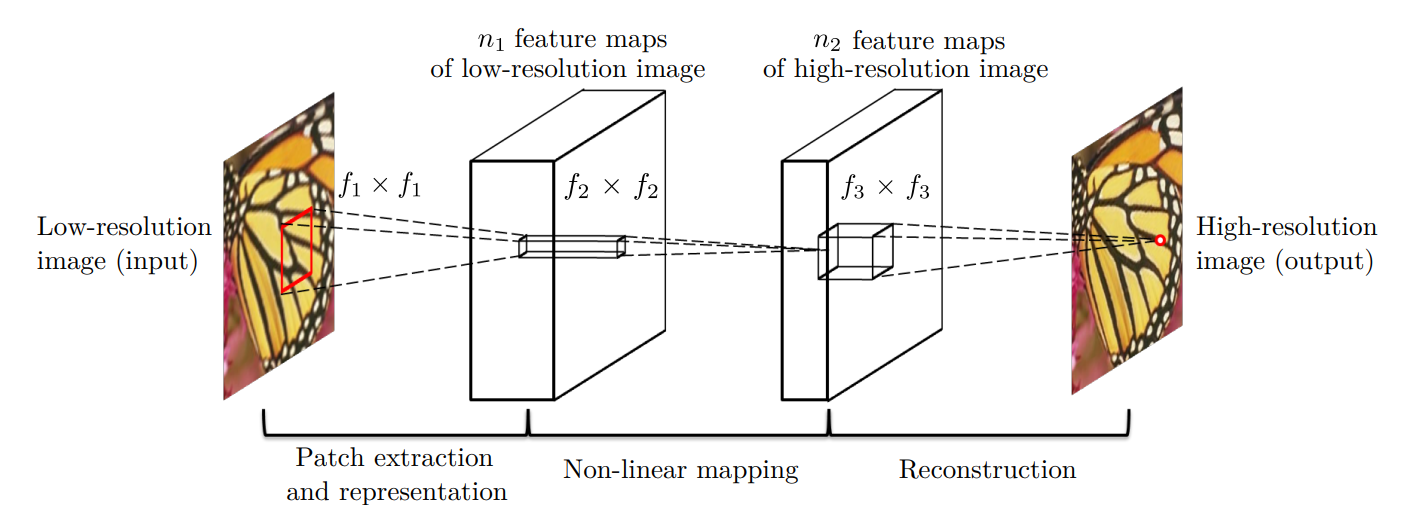
CNN在圖像相關的應用都有許多傑出的表現，像是這邊要提到的SRCNN（Super-resolution Convolution Neural Network），即為其中一個應用，Super-resolution，中文讀做超解析度，是一種提高影像解析度的技術，在之前，要對影像作提高解析度，通常只有透過插植才可以做到，但是用插植法插值出來的影像在細節度明顯不足，隨著CNN的研究深入，也有人將CNN應用於影像超解析度，用機器學習學習低解析度影像與高解析度影像間的映射關係，使得使用機器學習方法還原出來的高解析度圖像更為真實。

SRCNN的網路僅有三層CNN，分別對圖像做特徵抽取與表達、非線性映射及重建，大致結構如下圖所示

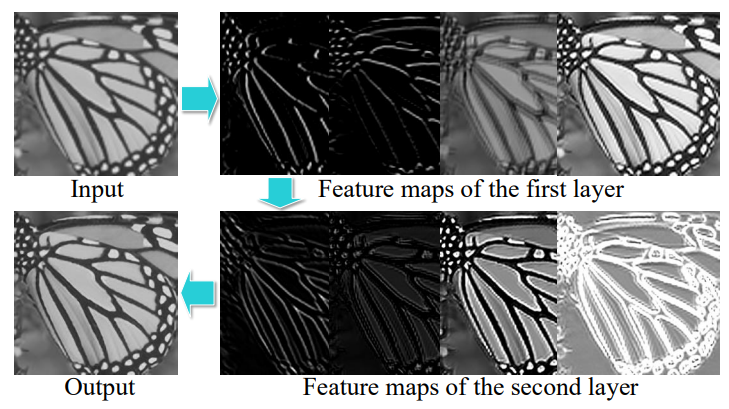
特徵抽取與表達層神經元數量計算為f2×f2×n1個，權重數量n1×(c×f1×f1+1)個。

非線性映射及重建層神經元數量計算為f3×f3×n2個，權重數量n2×(c×f2×f2+1)個。

非線性映射及重建層神經元數量計算為超解析度影像長寬乘積個，權重數量超解析度影像長寬乘積×2個。



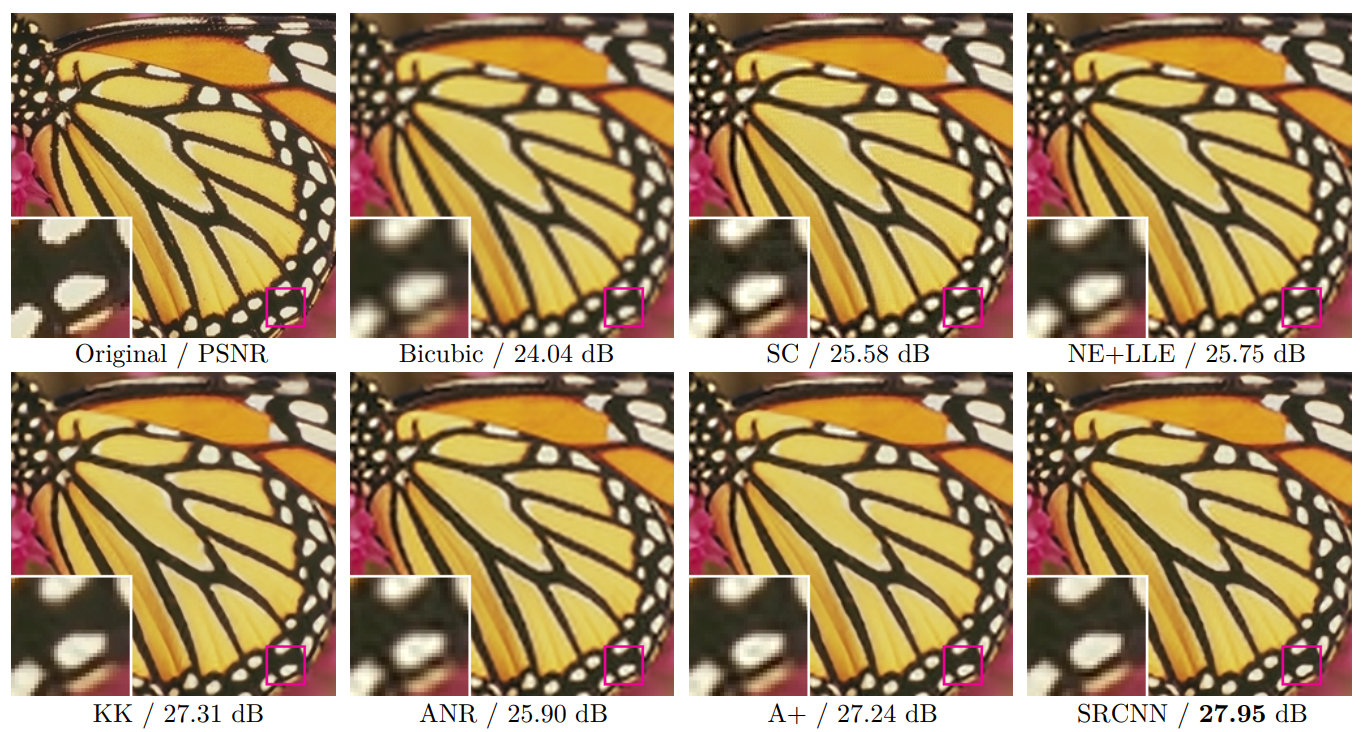
一張圖片經過網路的過程會像下圖所示。



在影像超解析度的不以正確率作效能的評估，最常見的是以還原資料及原始資料之間的峰值訊噪比（PSNR）作為衡量效能的數值，通常PSNR大於30，人類會開始分辨不出還原影像與實際影像的差異，從論文中沒有詳細提及整個訓練時間，不過有提到，如果要達到可以超越大多其他超解析度演算法的效能，大約要4\*108次backprops後才會有顯著差距，剩下的部分就取決於電腦的硬體。



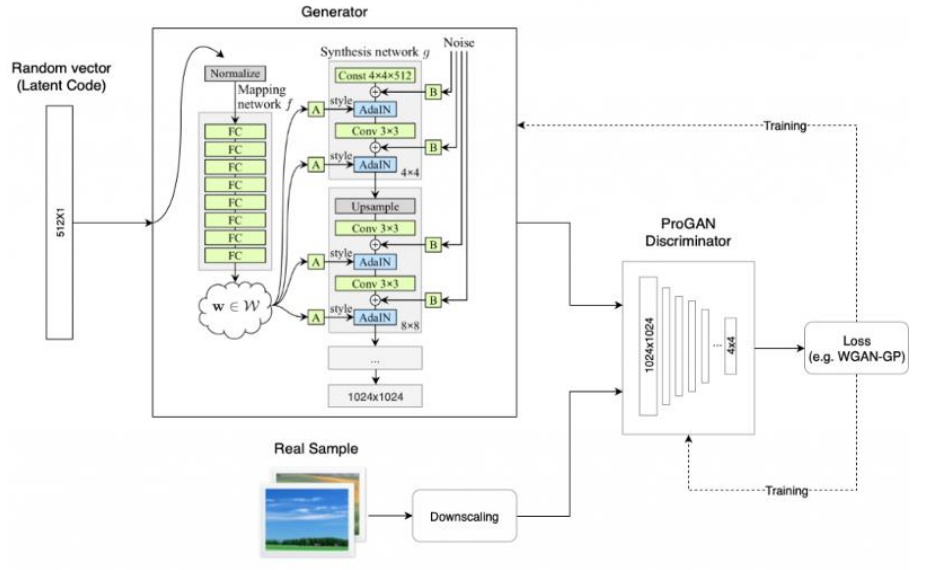
下圖可以清楚看出各超解析度方法的成效。



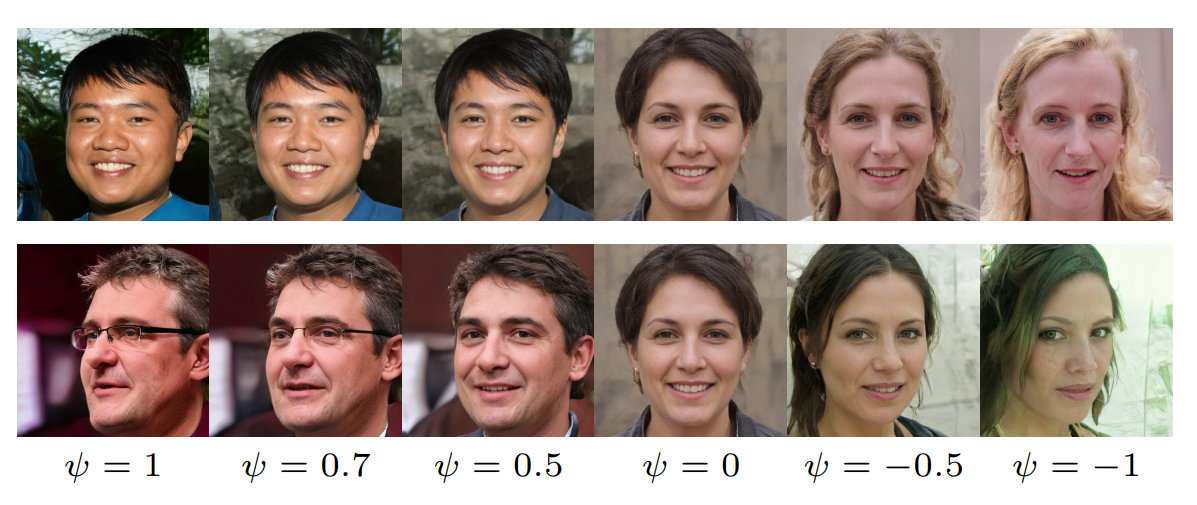
G. 近年來除了Convolutional Neural Network大放異彩以外，還有其它類型的Neural Network也大放異彩嗎？請你找出一個*其它類型的Neural Network*，並說明它在甚麼 應用方面大放異彩。它最近的結果(最好是最近一年內的結果)為何？它的NN有多少層？神經元有多少個？有多少個權重?正確率為何？訓練多久？...等相關資訊。

近年除了CNN外，還有一種特殊的Neural Network—GAN，全名生成對抗網路（Generative Adversarial Networks），一般用於生成以假亂真的圖片，以往的神經網路，通常都需要人來對資料做標籤，就如這次專案中的pattern，在訓練前都要給資料做標籤才能讓神經網路去判斷神經網路出來得結果好或不好，其原理簡單來說是訓練generator 及discriminator，generator需要訓練讓它生成假的資料，試圖騙過discriminator，discriminator則需要訓練到可以辨識圖片的真偽，兩個神經網路彼此互相訓練、調整參數，使用者需要給神經網路的，就只需要真實的照片即可，可以減少大量為資料上標籤的時間。

這邊要介紹的是StyleGAN，用於生成可以以假亂真的人臉，給網路一些Style，如人臉風格、表情、膚色等，就可以根據這些參數生成一些很像真人且符合標籤內容的人臉。



整體網路架構如上圖所示。



從論文中的結果可以看出對於一個參數ψ做調整可以看到在一個參數上對於輸出圖片連續性的變化。

如此龐大的網路訓練起來需要更多的時間，訓練時間部分根據該論文相關文件，用解析度1024\*1024的圖片資料集在使用單個Tesla V100 GPU需要訓練41天4小時，如此可見此神經網路之複雜。

H. 挑戰加分題(若有做，會加分)：EMNIST則為MNIST的擴充版，它增加了英文字母的手寫資料集。你有沒有辦法利用O'Neill先生所提供的專案，拿來執行一下英文字母 的辦識任務。，說明你如何執行訓練及測試？如何得出正確率？你自已是怎麼算的？ 這個訓練結果花了多少時間？請說明能做到好棒棒的辨識結果嗎？(請參考 https://jennaweng0621.pixnet.net/blog/post/403632155-mnist%E8%88%87emnist%E6%89%8B%E5%AF%AB%E8%B3%87%E6%96%99%E9%9B%86\_%E8%BC%89%E5%85 %A5%E8%88%87%E8%AE%80%E5%8F%96)。

I. 請說明你做此作業所踫到的一些狀況及困難。

本次作業是要完全使用由其他人開發的軟體進行操作，研究其網站來了解整個程式的運作，有些部分在網頁中也沒有寫的非常清楚，要多研究其他網站來大致了解，或是多與同學討論，看如何對程式進行操作，而且由於本程式是用cpu進行運算，所以只要在訓練時有作一些同樣需要大量cpu的功能，就會嚴重干擾訓練時間，使得訓練速度時快時慢，也沒有epoch次數的設定，所以無法良好的估計訓練時間，在其他神經網路應用的部分，大多的應用，都只有簡要的說明原理即大致的網路架構，如果要知道它內部詳細的參數，還需要去查詢更進階的資料，更有許多技術，是聽過，但是找不到相關論文、或是尚未公開的，所以在查詢資料上需要花上許多時間。

J. 請列出你的參考文獻(含網站)來源，並請說明參考了那些部份用於作業中。

- E題中關於Rosetta系統的介紹

<https://arxiv.org/abs/1910.05085>

- F題中關於SRCNN的論文介紹

Image Super-Resolution Using Deep Convolutional Networks

<https://arxiv.org/abs/1501.00092>

- G題中關於StyleGAN的介紹

<https://arxiv.org/abs/1812.04948>

<https://github.com/NVlabs/stylegan>