**專題計畫名稱 :**

**利用深度學習從衛星影像自動分辨建築物－以WHU Building Dataset為例**

Automatic building detection from optical satellite images

* Using Deep Learning

**專題研究生 : 楊尚峰**

**專題指導教授 : 林昭宏**

1. **摘要 :**

本專題研究目標為利用深度學習技術實作從衛星影像自動分辨建築物，主

要研究對象為WHU Building Dataset所提供的衛星影像，擷取影像中的建築物區域，進行辨識後得出是否為建築物之結果。

基於深度學習(Deep Learning)技術，本專題將使用卷積神經網路(CNN)提取特徵(feature)，並在卷基層、池化層嘗試不同的配置情況，以達到最佳的辨識率，主架構使用Autoencoder，最後評估實驗成果，彙整成結論。目前使用2700張訓練資料之模型的Mean square error可達到0.0153。在預測方面則有使用OTSU二值化算法尋找最佳閥值以及Dhash算法進行相似度分析。

**關鍵詞 : 衛星影像、深度學習、建築物辨識、Autoencoder**

1. **研究動機與研究問題 :**

基於本人就讀科系（國立成功大學測量及空間資訊學系）的專業領域課程－「遙感探測」，於課程中學習遙測及衛星影像等相關知識。課程中進行衛星影像中地物的監督式分類（supervised）與非監督式分類（unsupervised）實作，並體認到兩種方法的利弊，期望找出新的方法來達到提高分類精度與減少人工參與的時間耗損。近年來由於深度學習興起，並且已被大量應用到影像處理的相關領域，因此期望以深度學習的方式來進行衛星影像中目標物的辨識。

研究方向大致確定後需要進行題目篩選，衛星影像中常見的地物種類繁多，像是農地、聚落、湖泊等大型目標，亦可以是海岸線、道路等線性目標。

最終選定目標物為建築物，由於知名的志願性地理資訊平台「MapSwipe」。

「MapSwipe」是一個開放原始碼的軟體，[使用方式](https://www.youtube.com/watch?v=JwaKTMUyU7k)非常簡單，透過後端系統將衛星影像切好豆腐，透過手機介面讓全球志工挑選地圖上可能的道路或建築物。此計畫可以與 [OpenStreetMap的 Humanitarian Team](https://wiki.openstreetmap.org/wiki/Humanitarian_OSM_Team) 的 [Missing Maps Project](https://wiki.openstreetmap.org/wiki/Missing_Maps_Project) 合作，於大型災害發生時，透過更新的衛星資料，讓志工可以快速的區分出尚存的建築物，為地面救災隊伍提供更新過得地圖情報。

在使用這個App發現雖然有來自世界各地的志工可以進行建物的挑選工作，不過存在下列問題 :

1. 志工素質無法評估
2. 災難來臨時志工人數未必足夠，造成部分地區沒有更新資料
3. App本身並無相關獎勵機制，難以大量招募志工並提升知名度

基於上述「MapSwipe」的問題，我認為可以配合專題研究方向，使用深度學習以辨識衛星影像中的建築物，建立一套系統辨識所有的照片，以解決上述問題。

本研究中將以WHU Building Dataset中的建築物辨識資料為目標進行研究。深度學習的訓練資料以及測試資料均來自WHU Building Dataset。

**3. 文獻回顧與探討 :**:

**<激活函數>**

*(胡依淳, 2018)*在論文中提到，卷積神經網路中的關鍵在於卷積層的 kernel 權重值。權重值通常有三個改變方式，分別是 kernel size 、激活函數和卷積使用的 kernel 數量。對於kernel size則提到以奇數為主，所以輸入影像大小盡量不要是奇數為佳。ReLU (Rectified linear unit) 是目前比較受歡迎的激活函數，優點在於計算量小不涉及除法，一部分神經元的輸出為 0 造成了網路的稀疏性，並且減少了參數的相互依存關係，緩解了過擬合問題的發生，而且不會有非線性發生的梯度消失的問題。缺點則在於屏蔽太多，可能導致模型無法學習到有效特徵。

*(張凱勳, 2018)*也提到深度學習系統的關鍵特徵之一是使用非線性飽和函數(Rectified Linear Unit， ReLU)來替換其線性飽和函數(sigmoid、tanh)，優點在於解決梯度消失(vanishing gradient problem)，梯度消失為當神經元的梯度接近於 0 時，在反向傳播過程中接近於 0 的梯度會導致權重不會更新。此文章為第二篇推薦使用用ReLU的相關文獻，因此在本研究中會優先考慮使用ReLU。

**<一般化>**

*(Natalie Wolchover, 2018)*深度學習過程可分為兩個階段：時間較短的適配階段(fitting)與時間長得多的壓縮階段(compression)。在適配階段，網絡學習如何標記訓練資料；在壓縮階段，系統會精熟如何一般化，並以標記新測試資料的表現來衡量，而達到一般化才是深度學習的目標。在實驗中有許多狗的訓練照片，其中某些照片中有狗還有房子，其他照片則只有狗，經過實驗發現，在訓練的過程中系統會忘記狗與房子的關聯性，原因在於訓練到只有狗的照片時會抵消此種關聯性。有鑑於此，在挑選訓練照片時應該適度的挑選有些微差異性的訓練樣本以達到較好的一般化成果。

**<池化層>**

*(張耿豪 et al, 2018)*提到池化層運算主要採用最大池化(Max pooling)， 目的是將大規模的影像進行部分採樣(Subsampling)，將目標影像區分為若干區域矩陣，只保留子區域矩陣之最大值。不只能夠降低參數數量與可防止過擬合(Overfitting)，也擁有相當不錯的抗雜訊能力。經過其他文獻得知 [1]，除了Max pooling，還有average pooling、Overlapping pooling、Spatial Pyramid Pooling等其他方法，本研究應該會先以Max pooling為主，其他方法也會進行嘗試。

**<全連接層>**

*(張凱勳, 2019)*提到完全連接層(Fully Connected Layer)是將網路計算結果平坦化(flattening)，將多維度的特徵圖轉化為一維，變成基本的神經網路。並定義權重以進行每種參數的計算， 並根據誤差值更新參數，使得分類結果更準確。經過其他文獻得知 [2]，全連接層通常會不只一層，原因在於一層全連接層並不能有效的解決非線性問題。多一層全連接層就可以有更細部的特徵分類結果，提高辨識率。

**< CNN深度學習加速器設計技術>**

*(曾建霖 &郭峻因, 2018)*為了減少傳送資料所消耗的時間與能源成本，有一個常見的作法是直接將資料表示位元進行刪減（Word-Length Truncation），亦即原本的資料數據可能需要 32 個位元才可以表示，透過資料刪減技術，捨去較不重要的位元，用較少位元表示，如：16 位元、8 位元、4 位元⋯甚至是 1 位元來表示。計算出的結果會產生誤差，因為每筆資料能夠表示的精確度（Precision）已經降低，雖然誤差有可能會在 CNN 的運算過程中被消除，但是也可能導致最後的結果變成錯誤的，讓 CNN 做出錯誤的判斷。適當的位元刪減可以在不影響判斷結果的前提下減少資料量，因此只要是合理的使用刪減技術，確實是個減少資料傳輸負擔的有效辦法之一。由於硬體設備有限，並非每台研究用的電腦均有GPU配置提供的加速運算，因此考慮到訓練樣本數目，屆

時可能有必要進行資料量縮小的配置，本文的做法可供參考。

**<衛星影像分類上的應用>**

*(黃意文, 2018)*嘗試在分類地物時加入類神經學習方法，訓練資料為500筆，根據其實驗結果發現在經過類神經訓練之後的整體分類精度上升30%。作者也在文章中表明統計學上的區別分析理論認為，誤判地物的狀況最容易出現在不同類別的相交處，但是實驗中建築物有無類別相交處產生誤判的情況，原因並未說明。針對該點，個人推斷可能因為誤判地區的建物灰度值與建物平均灰度值差異過大而造成的誤判，應該屬於少數個案。該問題應該可以透過提高訓練資料數量來解決。

**<前處理>**

*(張家豪, 2018)*認為在資料進行訓練之前應該要做影像的前置處理以及資料資增強(Data Augmentation)來彌補其不足的特徵變化。以下是作者使用的處理方法。

1. 強度校正：利用 Histogram Equalization 避免影像過於偏重亮或是暗，使影 像的強度(亮度)有一個統一的標準。
2. 影像鏡像：隨機將影像產生其鏡像。
3. 影像旋轉：隨機對影像做小幅度的旋轉。
4. 影像裁切：隨機裁切原影像中的一部分作為新的影像，再將其拉伸為回原 始尺寸。
5. HSV 調整：對影像的色相、飽和度度、明度，進行隨機調整。
6. GAMMA 調整：對影像的 GAMMA 值進行隨機調整。

其中由於研究目標的差異，本專題應該不用做上面所有的前處理，可以嘗試進行的處理為第1、5、6項。

**<衛星影像辨識競賽>**

*(Geesara Prathap & Ilya Afanasyev ,2018)*是以參與深度學習的衛星影像建築物辨識競賽(SPACENET)為主題出發。作者提到前置作業使用2個標準差的歸一化處理，且影像來源除了RGB影像之外，還使用PAN(全色態)、MUL(8-Band Multi-Spectral images )等影像組成訓練和測試集。文中使用的深度學習網路是U-Net(用於生物醫學圖像分割的捲積網絡)，並且根據3種不同的影像來源做分別的3種訓練，最後再將3種成果平均後輸出，其成果與SPACENET前三名的差距在伯仲之間。作者也提出使用F score 來判斷模型的準確率，雖非原創但也不失為一個好方法，以及建議建造更深層的網路結構，甚至嘗試生成式對抗網路(GAN)以獲取較佳的結果。

[1] : 池化總結（OverlappingPooling、 一般池化、Spatial Pyramid Pooling） : https://codertw.com/程式語言/579142/

[2] : 卷積神經網路(CNN)入門講解: https://zhuanlan.zhihu.com/p/33841176

4. **研究方法與步驟 :**

考慮到首次建構的深度學習模型，可能不足以很好的進行模型訓練，本專題將嘗試採用Autoencoder的架構來建構模型，其中的CNN部分預計將以SegNet 為參考(SegNet架構圖見下頁)。

本次專題之研究架構中，分為encoder與decoder兩個部分。

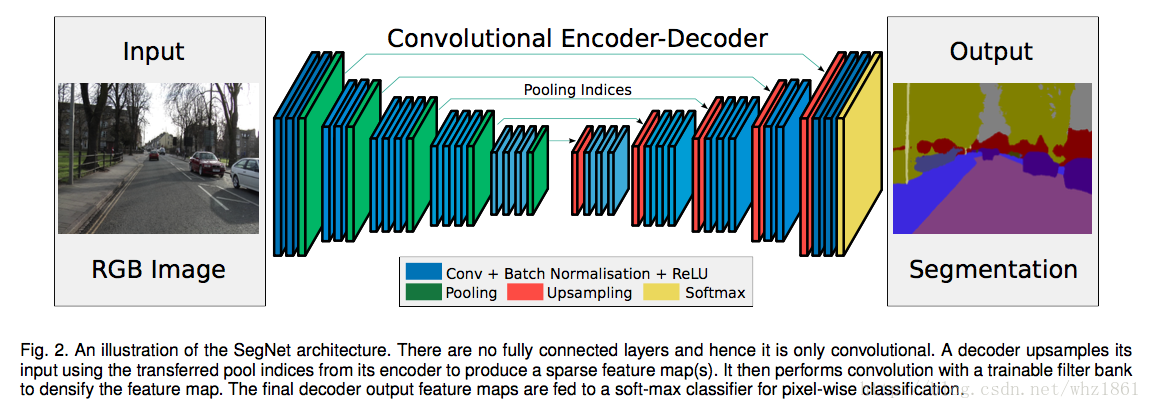
首先將資料集儲存成numpy的格式，之後經過影像前處理，接著進入encoder的部分。

Encoder : 透過 CNN 提取特徵圖(Feature Map)，並經由建築物分類器對該特徵圖計算出辨識結果，主要步驟為convoloution + maxpooling。最後得到該張影像的code(一張小圖片)。

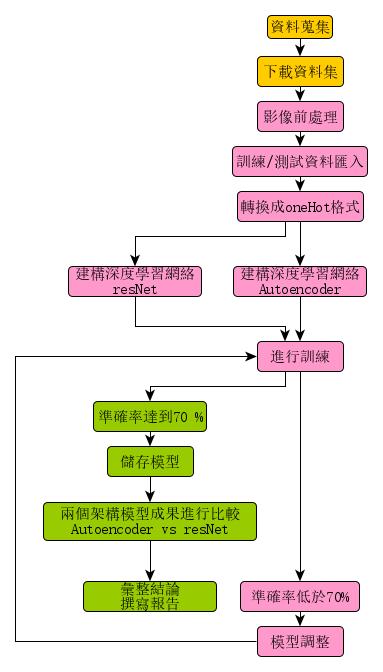
Decoder : 為了將code還原成初始大小，進行unsampling + deconvolution，且最終在每個像素都輸出一個結果－是否為建築物。

最後將結果與該張圖資的groundtruth資料比對後，將其結果作為loss function，並作出對整個模型的評估，藉此來訓練模型，以及測試整個模型的表現，以上為本專題的主要研究流程。

挑選SegNet作為參考架構的原因是其輸出影像以不同顏色區分不同物體，並且可以知道物體所在的位置，較適合用於衛星影像中區別不同的地物。而SegNet又可分為正常版與Bayesian SegNet，其差別在於Bayesian SegNet可以在進行預測之後提出分類不確定性的灰度圖，幫助改善不確定的區塊。預定先以SegNet為主要參考。詳細研究方法流程如下圖。



< SegNet 架構圖>



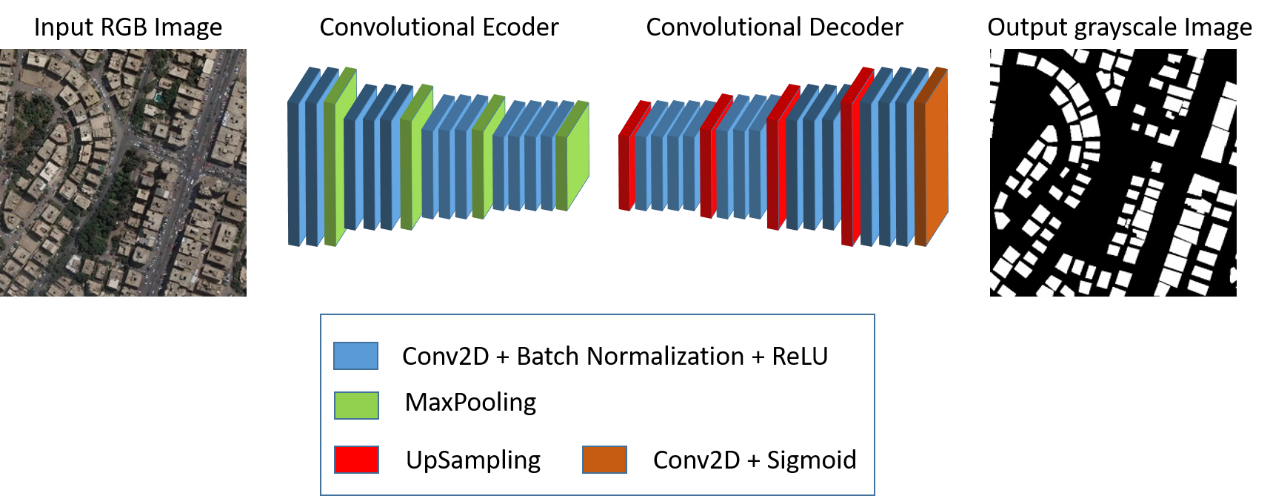
<研究流程圖>

**5. 預期結果 :**

建立一套以Autoencoder架構為主的深度學習系統來辨識WHU Building Dataset提供的測試資料，並期望達到辨識率超過70%，並嘗試建立另一套Deep Residual Net進行比較。

1. **自行設計網路之架構 :**

**CNN-Autoencoder**

****

模型基本參數

|  |  |
| --- | --- |
| CNN Layer filter size | 3\*3 |
| CNN Layer filter number | Encoder : 16->32->64->128 |
| Decoder : 128->64->32->16 |
| MaxPooling filter size | 2\*2 |
| UnSampling filter size | 2\*2 |
| Weight initializer | **he\_normal** |
| Optimizer | **Adam** |
| Loss | **Mean Square Error** |

**輸入 :**

256 x 256 大小影像(RGB)

**Encoder 部分 :**

由數個 [Convolution + batch normalization + ReLU ] + Maxpooling 組成。在Maxpooling 將影像縮小至32 x3 2時將不進行Convolution，而是再進行一次Maxpooling 來達到 bottleneck layer(16 x 16)。

**Decoder 部分 :**

照Encoder 部分的順序反向還原成256 x 256，最後以Sigmoid 來輸出是否屬於建築物的範圍，輸出影像為灰階影象。

**Loss 函數 :**

先以categorical\_crossentropy進行嘗試，後來採用Mean Square Error。

1. **研究資料說明**

WHU Building Dataset 有4個不同的dataset，此次研究主要使用Satellite dataset I (global cities)。

Satellite dataset I (global cities) :

1. 資料集大小 : 113Mb
2. 資料集數量 : 204張
3. 衛星影像大小 : 512 × 512
4. 衛星影像解析度 : 0.3 m to 2.5 m
5. 衛星來源 : QuickBird，Worldview系列，IKONOS，ZY-3等
6. 資料集包含之地區 : 武漢、台灣、洛杉磯、開羅、渥太華、威尼斯等
7. 挑選該資料集原因 : 204張測試資料除了不同衛星的傳感器差異之外，大氣條件，全色態和多光譜融合算法，大氣和輻射校正和季節的變化使這些測試資料適合測試建築物提取算法的通用性。

研究資料樣本(測試資料) 研究資料樣本(groundtruth)



Data Augmentation :

將原始資料(512\*512)**隨機切出4張**256\*256的圖片，再將這4張圖片進行**旋轉**(90-180-270) + **翻轉**(左右-上下)，完成後可得到4302張訓練資料/558張測試資料。

由於訓練時間&電腦效能因素，最後使用兩組資料進行訓練。

模型一 : 1080張training + 144張 testing

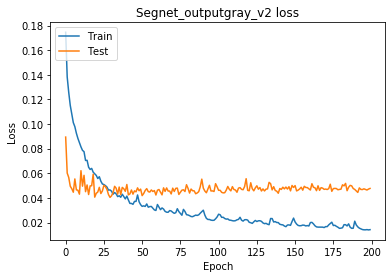
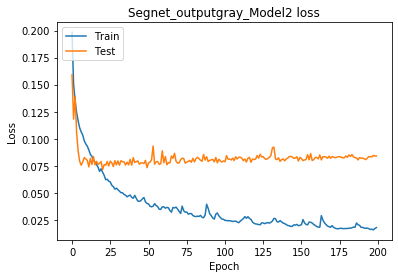
模型二 : 2700張training + 168張 testing

1. **模型成果**

模型訓練時間 :

模型一 : 約45分鐘 模型二 : 約2.5小時

模型一 模型二



模型一(下列4張影像預測時間:3.2605 sec)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| epoch | Batch\_size | mse | loss | val\_mse | Val\_loss |
| 200 | 16 | 0.0253 | 0.0253 | 0.0811 | 0.0811 |



模型二(下列4張影像預測時間:5.0491 sec)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| epoch | Batch\_size | mse | loss | val\_mse | Val\_loss |
| 200 | 16 | 0.0153 | 0.0153 | 0.0519 | 0.0519 |



**兩組模型差異**

1. 圖形完整度

由預測結果可知，模型一預測出的形狀較為破碎，而模型二預測出的形狀完整度較高。

1. 預測時間快慢

兩組模型在預測上的速度都很不錯，平均預測一張約0.8-1.26秒之間，雖然模型二參數數量較多，但速度只略遜於模型一。

1. 適合的預測對象

由預測圖形來看，模型一在小面積建物緊密排列時又良好表現(如預測圖最右邊)，反觀此種狀況對模型二來說，會傾向忽略中間為小的建物縫隙，產出趨於完整的圖形。模型二由於訓練資料較多，可適應較多種的建物種類，但是在細節的把握上有時不如模型一。

1. Overfitting 程度

無論是模型一還是模型二都可以見到有overfitting的現象，都是在訓練趨近收斂但尚未收斂時產生overfitting，相較之下模型二的overfitting現象較模型一來的改善許多。要解決overfitting問題需要由增加訓練資料數量/減少模型複雜度來達到，目前情況較適合增加訓練資料。本身模型有兩個任務 : 其一是將RGB轉為grayscale影像，其二是標示出建築物位置。因此模型任務算是複雜，不適合減少參數數量。

**嘗試 Retrain**

目前訓練資料為2700張，數量上仍屬於較少量，但是由於電腦設備的限制(gpu記憶體不足)，因此嘗試採取分段訓練的方式來增加模型泛化能力。具體步驟如下：

1. 訓練原始模型
2. 儲存model的h5檔
3. 匯入model並使其fit新的訓練資料
4. 以獨立的檢測資料進行model evaluation

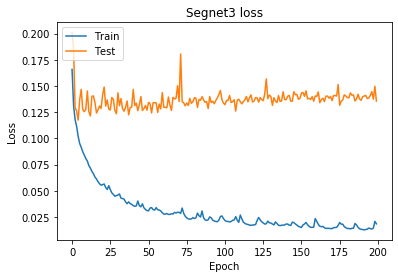
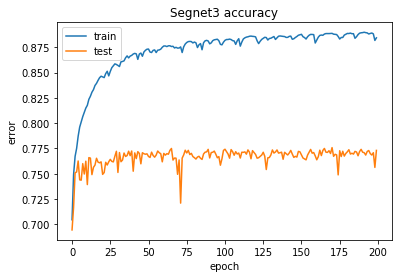
目前嘗試過兩次retrain

|  |  |  |  |
| --- | --- | --- | --- |
| model | Train\_data(張) | Valdation\_split | File\_name |
| Segnet3 | 3400 | 0.15 | Random\_new |
| Segnet3\_retrain | 1600 | 0.15 | Not\_train |
| Segnet3\_retrain2 | 3102 | 0.15 | Retrain3 |

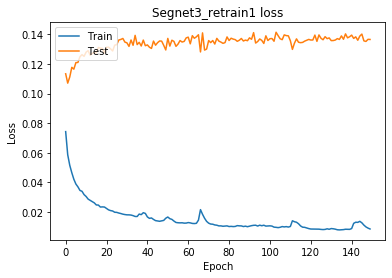
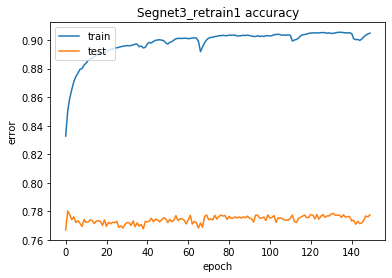
Epoch / batchsize

1st : 200 /16 2rd : 150 / 16 3rd : 200 / 16

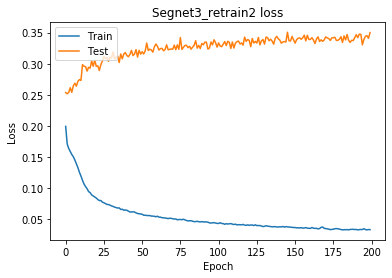
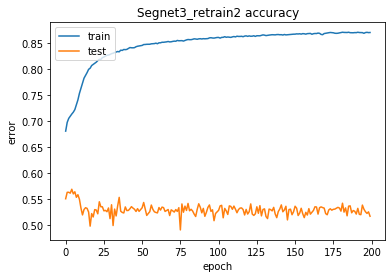
Segnet3



Segnet3\_retrain



Segnet3\_retrain2

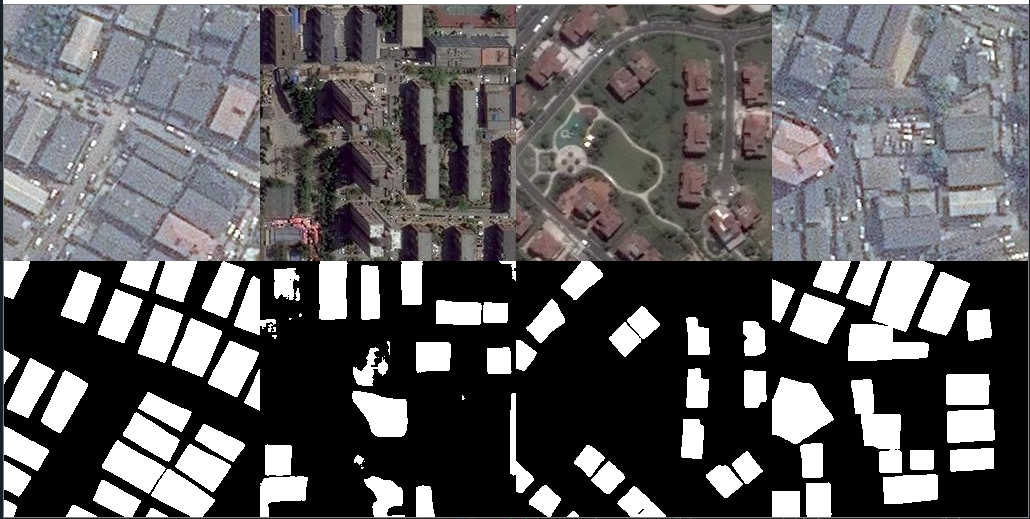


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | loss | Vla loss | Binary acc | Val Binary acc |
| Segnet3 | 0.0184 | 0.1335 | 0.8844 | 0.7731 |
| Segnet3\_retrain | 0.0084 | 0.1365 | 0.9049 | 0.7775 |
| Segnet3\_retrain2 | 0.0328 | 0.3501 | 0.8702 | 0.5174 |

-------------------------- evaluation(test,test\_labels) : 490張--------------------------------------

|  |  |  |
| --- | --- | --- |
| Model | MSE | Binary acc |
| Segnet3 | 0.09299260398913775 | 0.8608522 |
| Segnet3\_retrain | 0.08769400477890045 | 0.8694602 |
| Segnet3\_retrain2 | 0.13560464515561058 | 0.80908304 |

Segnet3\_retrain模型(下列4張影像預測時間:3.673 sec)



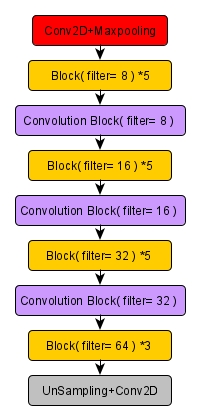
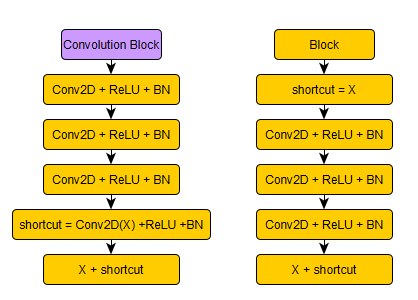
預測時間較模型二快上不少，除了第二張預測結果較差以外，其餘圖形皆十分完整。

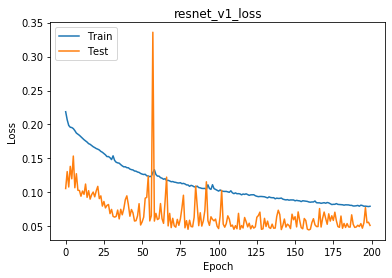
**總結**

從上述資料可以發現，在第一次的retrain後，不論是在loss或是accuracy方面都有小幅進步。但是在第二次增加3104張資料進行retrain候可以發現model有明顯的退步，推測原因是此次增加的資料與先前資料差異過大，導致參數更新幅度大，進而造成明顯的overfitting問題(只fit 這3104張)。要解決此問題方法有三，其一是升級電腦設備(提高gpu記憶體及其算力)，此舉可以增加一次訓練的資料量，減少retrain次數，即可一定程度避免資料差異過大。第二種方法是在training時不停變換training data set使模型增強泛化能力。最後一種方法是在Data Augmentation時要使製造出的新data 足夠隨機，亦及必須再進行旋轉/翻轉/放大/明暗度變化時加入一些機率的條件。

1. **與設計之ResNet之比較**

**ResNet架構 ResNet Block 設計**

****

****

模型三(下列4張影像預測時間:10.5411 sec)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| epoch | Batch\_size | mse | loss | val\_mse | Val\_loss |
| 200 | 10 | 0.0718 | 0.0718 | 0.0597 | 0.0597 |



模型訓練時間 : 約11小時

與自行設計之Autoencoder相比，首先在訓練時間上就有十足的差異，Deep Residual Net在訓練時間上是Autoencoder的5倍以上。在預測時間上也是Autoencoder的兩倍。

其次是精確度，以結果來說相比Autoencoder來的好上不少，但若以預測圖形來看，其產出的圖形較為破碎，接近模型一的情況，且易產生微小的失誤雜訊(如下圖)。



1. **預測處理**

**Threshold 處理**

雖然模型輸出的是grayscale image，但其輸出結果並沒有較明確的

黑白分界，亦即有部分模糊空間。為了改善此問題並提升輸出成果，將採用threshold二值化處理。Threshold二值化處理在OpenCV中有許多種類，指導教授建議使用Otsu Binary算法，因此一開始選擇以此法進行預測。

**OTSU Binary算法**

大津法（OTSU）是一種圖像二值化分割閾值的算法，也稱最大類間差法，有時也稱之爲大津算法，由大津於1979年提出，被認爲是圖像分割中閾值選取的最佳算法，計算簡單，不受圖像亮度和對比度的影響，因此在數字圖像處理上得到了廣泛的應用。它是按圖像的灰度特性，將圖像分成背景和前景兩部分。

背景和前景之間的類間方差越大,說明構成圖像的兩部分的差別越大,當部分前景錯分爲背景或部分背景錯分爲前景都會導致兩部分差別變小。因此，讓類間方差最大的分割意味着分錯概率最小。算法如下:

對於圖像 I(x,y)，

前景(即目標)和背景的分割閾值記作 T，

屬於前景的像素點數佔整幅圖像的比例記爲 ω0，其平均灰度 μ0。

背景像素點數佔整幅圖像的比例爲 ω1，其平均灰度爲 μ1。

圖像的總平均灰度記爲 μ，類間方差記爲 g。

假設圖像的背景較暗，並且圖像的大小爲 M×N，圖像中像素的灰度值小於閾 值T的像素個數記作 N0，像素灰度大於閾值T的像素個數記作N1，則有：

ω0=N0/ M×N (1)

ω1=N1/ M×N (2)

N0+N1=M×N (3)

ω0+ω1=1　 (4)

μ=ω0μ0+ω1μ1 (5)

g=ω0(μ0-μ)^2 + ω1(μ1-μ)^2 (6)

將式 (5) 代入式 (6) ,得到等價公式:

g=ω0ω1(μ0-μ1)^2 (7)

用窮舉法的方得到使類間方差 g 最大的閾值 T ，即爲所求。

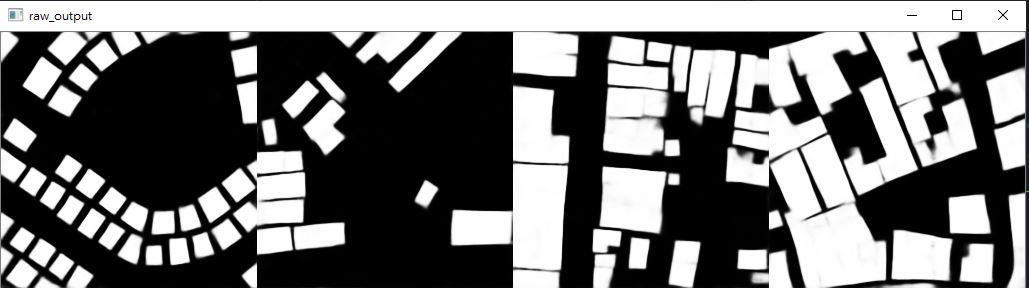
**OTSU Binary算法於專題中的缺陷**

當圖像中目標區域和背景區域的像素數量相差太大時 ,表現在直方圖上就是一個波峰要比另一個波峰大得多 。此時 ,若利用OTSU準則來求取門限 ,則由於原理的侷限性 ,所得到的門限不可避免地向較大的波峰方向“漂移”，導致目標區域分割的失敗。簡單來講，如果目標與背景所佔的區域大小相差懸殊，就不能有效的分割出目標區域。

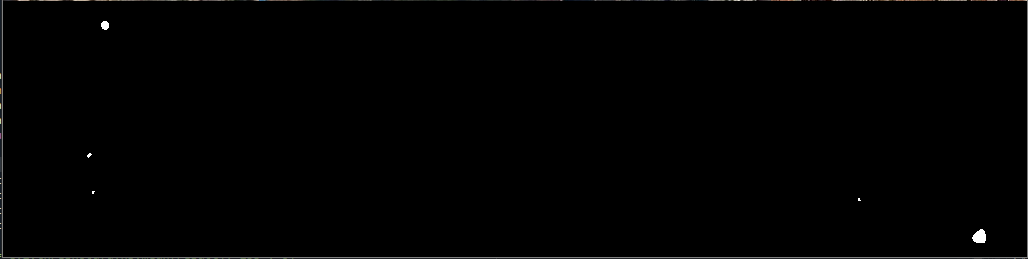
在部分測試資料中發現此問題，預測圖片基本上已不可辨識。或許簡單的一

次OTSU分割可能不能達到理想的效果。

model預測出來的raw image(大致已有分類好，但是有些地方是模糊的)

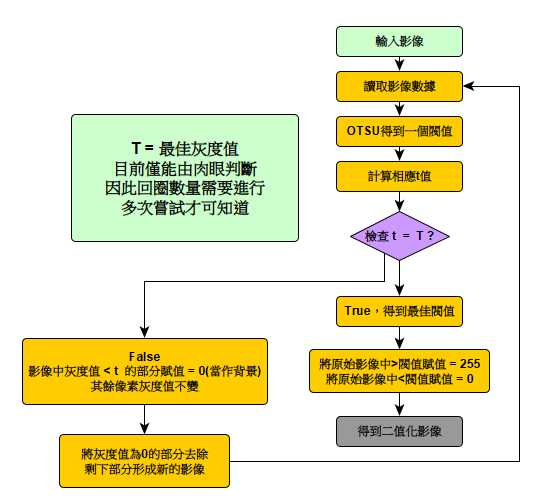


若採用一次OTSU則會造成以下結果



**解決方法**

已知分割結果圖象可以分爲兩部分:背景區域和(含有部分背景的)目標區域，由於我們只對目標區域感興趣，把背景區域去掉並不會丟失目標信息。我們可以對**預測圖像但是含有部分背景**的目標區域進行OTSU分割，得到新的預測圖像，同樣，把新的預測圖像中背景區域去掉，如果目標區域還包含有部分背景，那麼，我們再對（含有部分背景的）目標區域進行 OTSU分割，就這樣依次不斷遞歸下去，直到分割圖像達到理想的效果爲止。遞歸法將使用迴圈控制OTSU分割次數。程式流程如下:



**成效**

迴圈參數太過浮動，且需要針對每張測試圖片進行數次嘗試才可決定，不符

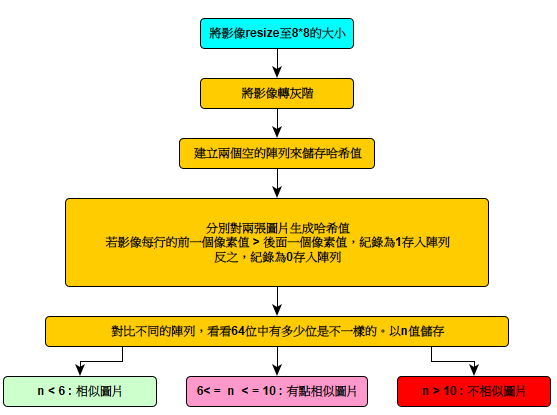
合時間成本，因此決定這些測試資料的閥值直接使用本身的平均灰度值，平均灰度值較為方便計算且成效還不錯。

對於OTSU預測較好的測試資料而言，OTSU給出的閥值與測試資料本身的平均灰度值相差在20個grayvalue之內，肉眼基本上較難看出差別，因此最後決定在全部的測試資料上的閥值都設為平均灰度值。

**量化預測結果**

預測出圖形後肉眼可以做初步的判斷分類結果，但是並不夠精確，需要一套

量化預測圖型與ground truth 之間相似程度的算法。本次專題採用感知哈希算法(perceptual hash algorithm)中的Dhash算法，具體步驟如下:



隨機檢測20張預測資料與其ground truth，結果如下:

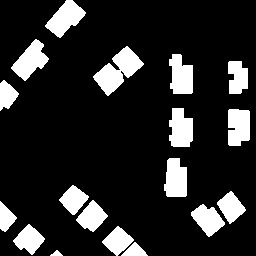
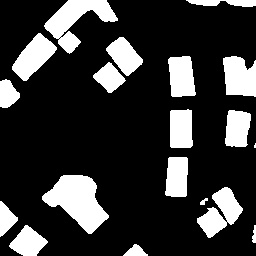
|  |  |  |  |
| --- | --- | --- | --- |
| 檔案名稱(.jpg) | n值 | 檔案名稱 | n值 |
| 406\_90 | 6 | 18\_90 | 3 |
| 421\_90 | 4 | 42\_90 | 13 |
| 428\_90 | 6 | 50\_90 | 16 |
| 469\_90 | 4 | 66\_90 | 12 |
| 35\_90 | 17 | 74\_90 | 15 |
| 59\_90 | 12 | 98\_90 | 8 |
| 75\_90 | 7 | 114\_90 | 11 |
| 83\_90 | 17 | 120\_90 | 15 |
| 2\_90 | 10 | 122\_90 | 9 |
| 10\_90 | 5 | 130\_90 | 20 |

平均n值 = 10.3(趨近於有點相似)

以平均n值來說還算不錯，但是仍要探討各種n值的影像有何特性。

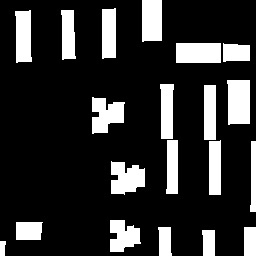
以10\_90.jpg(n = 5)為例:

預測 groundtruth



以75\_90.jpg(n = 7)為例:

預測 groundtruth



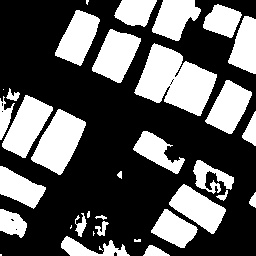
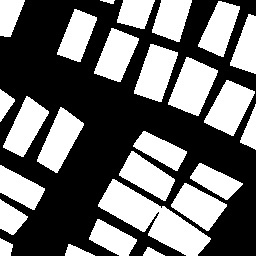
若單純從預測結果來看這張jpg的n值應該蠻大的，於是我將8\*8大小的照片取出觀看



若從resize的圖來看，確實有點相似，如圖中箭號處。Dhash算法在此張圖片明顯失效，具體原因推測為內差模式造成(目前採用INTER\_CUBIC，4x4像素的雙三次差值)。4x4大小的範圍差值對建物資料可能不夠敏感。

以42\_90.jpg(n = 13)為例:

預測 groundtruth

****

可見基本輪廓大致相同，有較大差異的地方在於圖中紅圈處，預測圖形明顯有2棟建物未能成功分類。在計算哈希值時，應該是此處的差異造成

**n值上升**。因此讓n值降低的關鍵在於要分類出所有建物輪廓，不能缺失。

以130\_90.jpg(n = 20)為例:

預測 groundtruth

****

圖中蠻多細節缺失(中間部分及右上半部)，因此n值高是可以預期的。

1. **結論與建議**

**結論**

* 1. Autoencoder精確度中上但耗時短，適合進行初步辨識建物
  2. ResNet精確度高但耗時太久，且對電腦效能要求較高
  3. 完全使用CNN Layer將使模型保有彈性，接受任意尺寸影像輸入
  4. 模型預測上使用Otsu算法來決定閥值，可以提升模型分類成果，但在部分建物面積與背景差異過大的圖片上效果較差
  5. 考慮計算便利性與分類成效，最後閥值均以平均灰度值為主。
  6. 建物分類難處在於建物形狀多變，且密集程度差異巨大
  7. 訓練遇到OOM(Out of Memory)情況時，可嘗試減少Layer以及Batch size
  8. 輸出層使用sigmoid函數適合二分類問題，也是影響速度因素之一
  9. Dhash算法可以快速判斷預測圖片與groundtruth的差距，不過需要考慮內差模式
  10. Retrain model的新增資料不能與先前資料差異過大，也需考慮Data Augmentation的隨機性

**建議**

1. 未來可以增加訓練/測試資料數量，目前只有2700張稍嫌不足
2. 增加Data Augmentation 方式(如調整亮度/透鏡畸變/Scaling等)，尤其需加入機率條件來增加隨機性
3. 未來可嘗試對每個pixel做multi-hot encoding 後進行分類
4. **參考文獻**

1. 林柏志，2014，利用2階段分類法來提高衛星影像中休耕田偵測之精準度，逢甲大學都市計畫與空間資訊學系碩士班碩士論文

2. 黃意文，2006，知識擷取與模糊推論在衛星影像分類上的應用－以台中都會區為例，朝陽科技大學環境工程與管理系碩士論文

3. 董炤巖，2010，以物件導向分類法進行 SPOT 衛星影像之崩塌地萃取，國立屏東科技大學森林系碩士學位論文

4. 王彥幃，2012，空間相干法在衛星影像地物分類之應用，國防大學理工學院環境資訊及工程學系空間科學碩士班碩士論文

5. 張家豪，2018，基於深度學習技術之行人年齡與性別辨識，國立交通大學資訊科學與工程研究碩士論文

6. 張凱勳，2019，植基於深度學習之影像辨識技術之研究，國立虎尾科技大學資訊工程系碩士班碩士論文

7. Natalie Wolchover，2018，學習的奧祕是遺忘－揭開深度學習神秘黑箱的新理論，DOI :10.6851/MSHCM.201801\_(13).0007

8. 郭宗杰(C.-C. Jay Kuo) & 趙學信，2016，關於深度學習網絡的兩個問題－為什麼需要深度網絡與非線性機制，DOI :10.6851/MSHCM.201610\_(10).0008

9. 張耿豪、蔡雅惠、呂寧遠、李韋辰、鍾幸芸、賴昱廷，2018，人工智慧發展與工業影像檢測新方法，智慧製造工業聯網技術專輯

10. 曾建霖、郭峻因，2018，淺談CNN深度學習加速器設計技術，國立交通大學電子研究所，DOI：10.6328/CIEE.201809\_(3).0003

11. 胡依淳，2018，深度卷積神經網路中卷積層之分析及比較，國立暨南國際大學電機工程學系碩士論文

12. 張家豪，2018，基於深度學習技術之行人年齡與性別辨識，國立交通大學資訊科學與工程研究所碩士論文

13. Geesara Prathap & Ilya Afanasyev，2018，Deep Learning Approach for Building Detection in Satellite Multispectral Imagery，Innopolis University(Russia)

備註:

參考文獻1~6來源為台灣碩博士論文加值系統

參考文獻7~12來源為華藝線上圖書館

參考文獻13來源 : https://arxiv.org/pdf/1811.04247.pdf

程式碼連結 : <https://github.com/GMfatcat/satellite_image-autoencoder.git>