

# 以CNN為基礎發展之卷積神經網路模型 運用於口罩瑕疵檢測之真實案例探查

楊永聖<sup>1,a</sup>, 何承遠<sup>1,2,b,\*</sup>

<sup>1</sup> 亞洲大學資訊工程學系

<sup>2\*</sup> 亞洲大學人工智慧博士學位學程

<sup>a</sup>ams7758@gmail.com, <sup>b</sup>tommyho@asia.edu.tw

## 摘要

2019年Covid-19病毒大流行，全世界的人們紛紛戴上了口罩，一時之間口罩變成了出門必備的東西之一，但疫情剛開始流行的時候因為民眾瘋狂搶購導致口罩大缺貨，也因此口罩的需求提升使許多工廠擴增產線或加入生產口罩的行列，但因為生產口罩時有許多因素會導致口罩有各種瑕疵出現，所以需要透過人工目視的方式檢查口罩是否有瑕疵，但人工目視會隨著時間產生疲勞，同時人力成本也相當的高，因此本論文希望藉由人工智慧取代使用人工進行檢查的部分以此節省大量的人力成本，使用人工智慧進行瑕疵檢測在產線中可以節省大約1~2個人力，且使用一台電腦進行瑕疵檢測系統可以同時對數百張圖片進行檢測，所以在工廠中可以降低不少人力成本。

本研究使用卷積神經網路中常見的五種模型針對口罩瑕疵檢測進行研究，共使用11,960張口罩圖片分別進行訓練、驗證以及測試，其中用來做最終準確度測試的圖片為隨機取出的1,198張圖片，實驗結果顯示最好的模型和其準確度為InceptionResNetV2和99.58%。

**關鍵詞：**人工智慧、卷積神經網路、瑕疵檢測、口罩、智慧製造。

## 1. 前言

2019年新冠肺炎爆發，幾乎全世界受到疫情的影響，但因此也催生了許多應用人工智慧的服務，使其人力需求減少進而減少感染與散播的機會，這些人工智慧服務與應用都是在日常可見的地方[1]像是出入口的體溫監測、無人車、無人工廠等，上述的前兩項應用在生活中非常普遍，但無人工廠的應用隨著每個工廠的產品與需求不同，會產生各式各樣不同的需求，加上疫情的影響讓口罩成為了人們出門的必需品之一，目前口罩在製造的時候還是仰賴人力進行品質管控，但使用人力進行品質管控會隨著時間累積疲勞產生疏失進而導致瑕疵口罩的數量增加使產能降低，且儘管在生產的時候人員進行消毒並且做好

保護措施還是有可能因為疏忽導致口罩被污染，所以要使產線轉型成無人產線的話就必須利用人工智慧進行升級，因此本論文嘗試使用深度學習對工廠中的口罩進行瑕疵檢測，期望能透過人工智慧辨識在產線上的口罩是否有瑕疵並減少品管人員的使用。

本論文為了要取得瑕疵口罩的影像資料，本論文找到了裕發科技股份有限公司[2]合作，提供本研究團隊瑕疵口罩與相關的器材，但資料集的部分是由本論文根據裕發科技股份有限公司提供的經驗與意見進行分類並拍攝製作而成，此部分將在研究方法中詳細說明。

近年來，隨著人工智慧的發展[3~6]，深度學習越來越被廣泛的應用[7]，尤其是卷積神經網路(Convolutional neural network, CNN)在解決影像分類問題的部分是有目共睹的[8]，本論文想利用深度學習可以自己學習到特徵的這個特點來達到類似自動光學檢查(automatic optical inspection, AOI)瑕疵檢測的效果[9]，所以本論文挑選了近年來在ImageNet上表現不錯[8,10]的五種模型來當作瑕疵檢測的模型，嘗試使用這些模型在本論文的瑕疵口罩資料集中也能有不錯的表現，並期望在未來中可以實際應用在工廠中。

## 2. 研究方法

本論文的研究流程分為三個部分如圖1所示，首先是收集資料，在取得口罩之後將口罩進行分類，並對口罩分別進行拍照取得初步的資料，接著是資料預處理，將收集的資料處理過使其在神經網路中能更好的被訓練，同時可以對資料進行擴增，使模型的準確度更高，再來是將資料放入模型中進行訓練，並比較各個模型的在測試資料中的效果，最後挑選效果最好的模型作為瑕疵檢測系統。

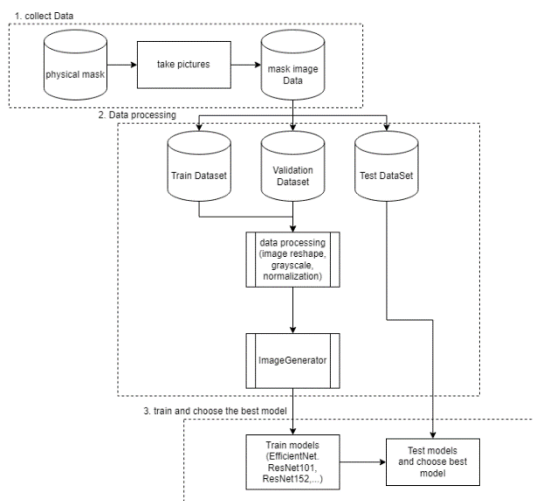


圖 1. 流程圖

## 2.1 研究設備與環境

本論文使用一台桌上型電腦作為實驗設備，其電腦硬體配置與環境配置如表 1，該實驗在 Ubuntu 20.04 環境中使用 Jupyter NoteBook 來進行 python 程式撰寫，搭配 tensorflow-gpu 使用 GPU 來為模型進行訓練以提升訓練與預測速度。由於 Ubuntu 系統本身不包含 Tensorflow-gpu、cuda、cudnn 等軟體，所以這些軟體必須額外安裝，在安裝 tensorflow-gpu 之前需要先安裝好 Nvidia cuda 以及 cudnn，且這兩者的版本需與 Tensorflow-gpu 的版本匹配才有辦法使用，詳細資訊可以參考 Tensorflow 官方網站與 Nvidia 官方網站。

表 1. 電腦環境配置

硬體配置	
CPU 處理器	Intel i5-9400F
RAM 記憶體	32GB
GPU 顯示卡	Nvidia RTX3060 *2
系統/軟體配置	
作業系統	Ubuntu20.04
Tensorflow_gpu	2.4.0
CUDA	11.0
cudnn	8.0

## 2.2 資料收集

為了要取得大量的瑕疵口罩與正常的口罩，本論文找到了裕發科技股份有限公司提供本論文各種類型的瑕疵口罩，使用裕發科技股份有限公司生產的口罩與瑕疵口罩進行分類並製作成

一個瑕疵檢測資料集，本論文將口罩分成四個類別，分別是正常、鐵絲瑕疵、耳掛斷裂、不良品，將每個類別的口罩分開並分別進行拍攝，其分類標準如表 2 所示，最終瑕疵檢測資料集一共收集了 11960 張圖片，再將這些圖片再進行拆分，分成訓練、驗證以及測試三個資料集用來訓練以及測試，其拆分的比例使用 7:2:1 對各類進行拆分。



圖 2. 不良品



圖 3. 耳掛斷裂



圖 4. 鐵絲瑕疵



圖 5. 正常

表 2. 口罩瑕疵分類標準

類別	不良品	耳掛斷裂	鐵絲瑕疵	正常
瑕疵分類標準	口罩外觀有瑕疵、布料不平整，但不影響口罩功能。	口罩耳掛斷裂或沒有耳掛。	鐵絲位置偏移、錯位、斷裂、太長或沒有鐵絲。	沒有瑕疵，能正常使用的口罩。

### 2.3 資料預處理

在訓練瑕疵檢測系統的模型之前，為了能讓模型能夠有效的學習到正確的特徵，所以會先將圖片進行處理再進行訓練，此外，本論文為了模型能夠更好的被訓練，所以本論文使用了 TensorFlow 中 ImageDataGenerator 的方法對資料進行擴增，使圖片從 11,960 張擴增到四十萬張以上，以此來讓模型能夠更好的被訓練。

處理資料的第一步是先將圖片縮小，因為本論文拍攝的資料集中圖片的大小是 1920x1080，所以為了能夠讓模型可以更快的訓練，必須先將圖片縮小，本論文將圖片透過 ImageDataGenerator 的內建功能調整成 740x370，同時將口罩進行灰階處理，因為擔心模型將口罩的顏色當成一種特徵來學習，本研究的資料中也有些部份的口罩顏色沒有其他類別豐富，所以很可能被模型當成一種特徵來學習，除了上述的方法本論文還使用 rescale 的方法，將口罩的像素從 0~255 之間轉換成 0~1 之間的浮點數，避免模型在訓練的過程中因為像素的數值太大導致神經元死亡，以上動作都是在 ImageDataGenerator 中完成並加上 horizontal\_flip、width\_shift\_range、height\_shift\_range 等功能讓圖片進行水平翻轉與左右上下些微的移動讓模型盡量不要只學到固定位置的特徵。

最後再透過 ImageDataGenerator 中的亂數功能將訓練資料打亂，並設定好 batch\_size 等參數即可開始訓練並且調整訓練參數。

### 2.4 瑕疵檢測系統

本論文選擇了 ResNet101、ResNet152、InceptionResNetV2、InceptionV3、EfficientNet 等 5 種模型最為瑕疵檢測系統的模型進行訓練，這些都是較常見且在 TensorFlow 2.4 版中可以直接使用的模型。除此之外，EfficientNet 中還有 8 種網路深度、寬度、解析度不同的模型，所以本論

文先從這八種模型中挑選了以現有的設備能夠順利運行且表現得最好的 B3 當作代表與其他模型進行比較。

為了能夠讓訓練進行得更順利，本論文也將訓練參數進行了調整，在將圖片輸入模型的部分將 input\_layer 調整成 740x370 的大小讓圖片比一般的資料集大一些使模型可以看到口罩的細微的差異，但這個大小同時也會降低訓練速度與增加模型的複雜度，使部份的模型因 GPU 記憶體不足無法使用。

此外訓練前還有 earlystop、reduce\_lr、optimizers 三個參數，earlystop 是用來監控模型在訓練過程中是否有順利進行，如果出現無法收斂、過度擬合或收斂太慢等問題就會終止訓練，這邊本論文將學習率訂在只要超過 10 個迭代沒有改善超過 0.0001 就停止訓練。再來是設定 reduce\_lr，這個參數會在學習過程中監控設定的指標有無改善，如果沒有改善的話就會降低學習率來改善學習的過程，此外 earlystop 與 reduce\_lr 監控的指標都是驗證準確度，這是指本論文使用的三個資料集裡面的驗證資料集，在訓練過程中會用來驗證本研究學習的成果，最後就是 optimizers，這就是在設定前面所說的學習率，本論文將學習率定在 0.00025 而非常見的 0.0001 是因為模型在初期會有一段時間無法有效的學習，為了模型能快速的脫離那個階段不要觸發 earlystop 機制且不希望模型因為學習率太高導致模型學習的效果不好，所以將學習率定在 0.00025。

最後為了要讓模型能夠有更好的成果，本論文在模型的最後加了 BatchNormalization 層對數據做正規化，再用激活函數與扁平層讓模型降維，最後使用 Dropout 確保模型不會過度擬合再使用 Dense 層對最終的結果進行分類。

調整完參數之後再來就是調整模型，為了能讓模型有更好的成果，本研究在模型的最後增加了 BatchNormalization 層在最後的時候對數據做正規化，再使用激活函數與扁平層將模型降維，最後使用 Dropout 防止過度擬合與 Dense 層對最終的結果進行分類。

## 3. 研究結果

最終本論文訓練出來的模型驗證準確度如表 3 所示，在 EfficientNet 中是 B3 的準確度最高且表現最好，再將 EfficientNet 與 ResNet101、ResNet152、InceptionV3、InceptionResNetV2 相比，如表 4 所示表現最好的模型與這篇論文[10]在 ImageNet 中的表現不同，反而是 InceptionResNetV2 的表現較好，會得出這樣的結

果的原因可能跟本論文使用的影像不是常見的正方形圖片有關，ImageDataGenerator 中的亂數調整與資料擴增也會有些微的影響。

表 3. EfficientNet 之間的比較

模型	Efficient NetB0	Efficient NetB1	Efficient NetB2	Efficient NetB3
驗證準確度	98.11%	98.32%	98.61%	98.49%
測試準確度	96.32%	95.90%	95.90%	96.41%

表 4. 各模型之間的差異

模型	ResNet101	ResNet152	InceptionV3	Inception ResNetV2	EfficientNetB3
驗證準確度	81.41%	92.17%	96.98%	98.70%	98.49%
測試準確度	91.48%	83.13%	95.40%	99.58%	96.41%

本論文認為在靜止的狀態下拍攝並且辨識的準確度至少應該要達到 98% 以上才有取代品管的可能。雖然本實驗處於資料相對不足、口罩保存的環境太過潮溼等問題增加了本研究在收集資料與實驗部分的困難，但儘管如此還是有三個模型達到了 95% 以上的準確度，本實驗注重的是瑕疵與正常的口罩能不能被準確的分類出來，所以本論文將測試資料集的數據做成了 confusion matrix 如圖 6~圖 8 所示，可以看到不良品與鐵絲瑕疵的部分在 InceptionV3 與 EfficientB3 有不少的口罩被分類成正常，且最明顯的耳掛斷裂也有被分錯的情形，這種情況在實際情況中是絕對不允許的，所以可能還要從資料或者參數的部分進行微調，在 InceptionResNetV2 的部分雖然只辨識錯了五張，但也有可能只是對

這個資料集的部分準確度很高，未來可能還需要將模型加入產線實際測試才知道是否是真的準確度很高。

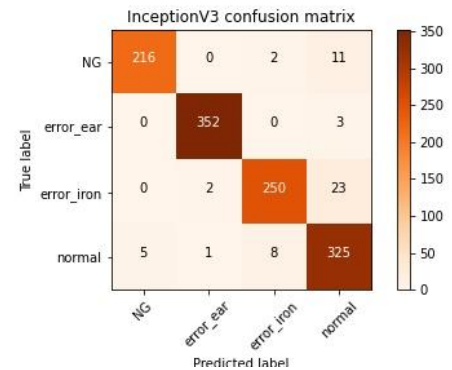


圖 6 InceptionV3 confusion matrix

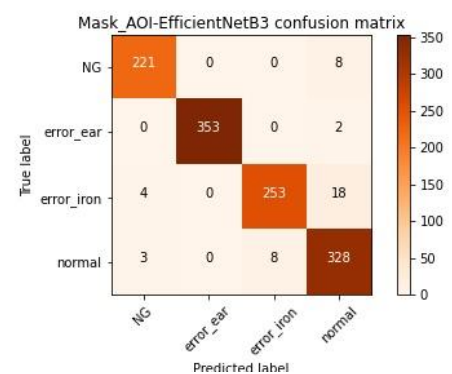


圖 7 EfficientNetB3 confusion matrix

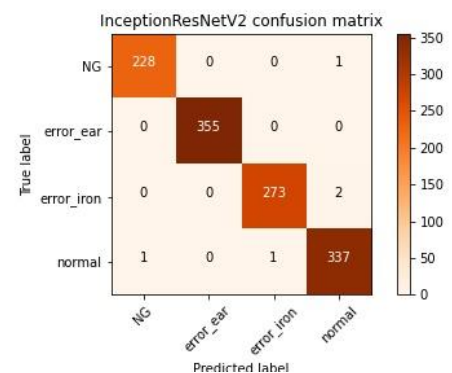


圖 8 InceptionResNetV2 confusion matrix

#### 4. 結論

經過實驗後發現，InceptionResNetV2 在測試資料集中取得最好的成績，InceptionResNetV2 與 EfficientNetB3 在測試資料集中的準確度差了 3% 以上，但在驗證資料集中表現卻沒有差的很多。這些差異雖然跟演算法本身有關係，但本論文在 ImageDataGenerator 中設置了亂數調整資料順序與將資料擴增，所以有可能會因為這樣造成模型每次學習的結果都不太一樣，且 EfficientNet 還有一半的模型沒有使用到，有可能使用 B4 或以上的模型結果會比 InceptionResNetV2 還要好。

使用 CNN 進行口罩的瑕疵辨識在靜止的狀態下的確可以有不錯的準確度，但還是會有瑕疵口罩被辨識成正常口罩，對於產線來說要在產線上使用還有一段距離，因為在實際生產的過程中會遇到各種問題，使用 CNN 進行口罩瑕疵辨識會遇到的問題可能有：

1. 產線中的口罩會不停的在移動。
2. 鏡頭無法跟上口罩移動的速度，導致捕捉到的影像變模糊。
3. 瑕疵檢測系統如何從模糊的影像正確地檢測口罩。
4. 如何讓口罩可以在進入鏡頭可捕捉的範圍的同時讓模型啟動並且檢測是否有瑕疵。

以上條列出的問題未來或許能透過物件偵測作為輔助來幫助瑕疵檢測系統能夠更快速地進行檢測，並使用更好的鏡頭以及更多的資料讓 CNN 可以更精準地捕捉到瑕疵，進而改善模型的準確度，以上將成為本研究團隊未來的研究方向。

## 致謝

本研究接受科技部編號：MOST 109-2221-E-468 -008 -MY3 研究計畫經費補助。

能夠完成此論文首先要先感謝合作廠商裕發科技股份有限公司提供資料、設備與經驗讓整個為本論文能夠順利的完成並取得初步的成果，再來要感謝朱學亭副教授在我的實驗遇到困難的時候提供不同的意見讓我可以解決實驗中遇到的困難，再來要感謝實驗室的王家恩同學幫忙分類並協助拍攝數以萬計的口罩，最後一併感謝所有提供意見的老師們，感謝他們在百忙之中給遇到問題的我許多不同的意見。

## 參考文獻

- [1] Bhuiyan, Md Rafiuzzaman, Sharun Akter Khushbu, and Md Sanzidul Islam. "A deep learning based assistive system to classify COVID-19 face mask for human safety with YOLOv3." 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 2020.
- [2] 裕發科技股份有限公司 <http://www.pumma.com.tw/>
- [3] LeCun, Yann, et al. "Backpropagation applied to handwritten zip code recognition." *Neural computation* 1.4 (1989): 541-551.
- [4] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." *nature* 521.7553 (2015): 436-444.
- [5] LeCun, Yann, et al. "Gradient-based learning applied to document recognition." *Proceedings of the IEEE* 86.11 (1998): 2278-2324.
- [6] Oh, Kyoung-Su, and Keechul Jung. "GPU implementation of neural networks." *Pattern Recognition* 37.6 (2004): 1311-1314.
- [7] Bengio, Yoshua, Yann Lecun, and Geoffrey Hinton. "Deep learning for AI." *Communications of the ACM* 64.7 (2021): 58-65.
- [8] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012).
- [9] Hecht, Oded, and Giora Dishon. "Automatic optical inspection (AOI)." *40th Conference Proceedings on Electronic Components and Technology*. IEEE, 1990.
- [10] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.