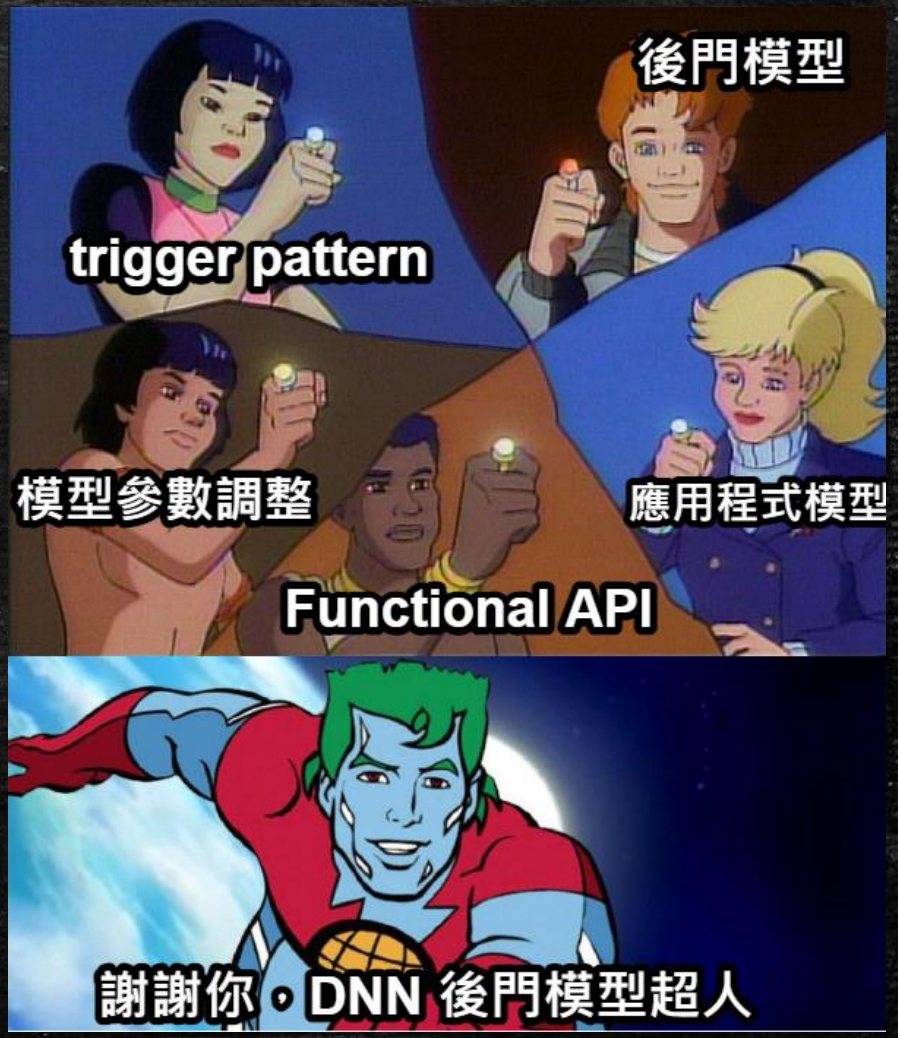
2024 鐵人賽 – 我數學就爛要怎麼來學 DNN 模型安全
Day 17 – 製作 DNN 模型後門(模型篇)

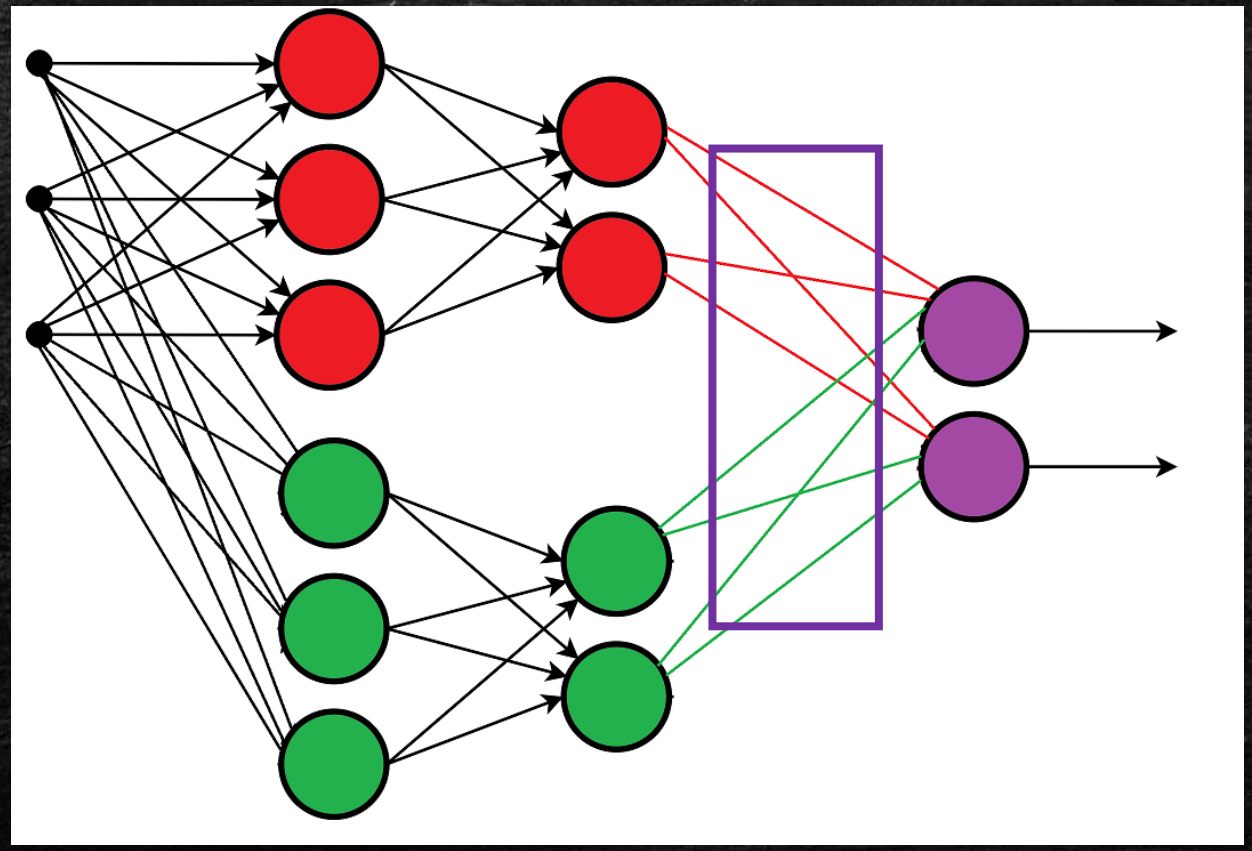
大綱

- 製作 DNN 模型後門 (模型篇)
  - 前情提要
  - 程式實作
- 結論



前情提要

- 把兩個模型輸出接在一起，之後接到一個輸出層，接著調整紫色框框的權重讓其生效



程式實作

- 上次已經先做定義 trigger 是一個 + 符號，並且訓練一個後門模型去辨識該 trigger
- 綠色的應用程式模型沿用之前的例子即可
- 接下來問題是要怎麼串接兩個模型的輸出，以及如何調整參數

Concatenate layer

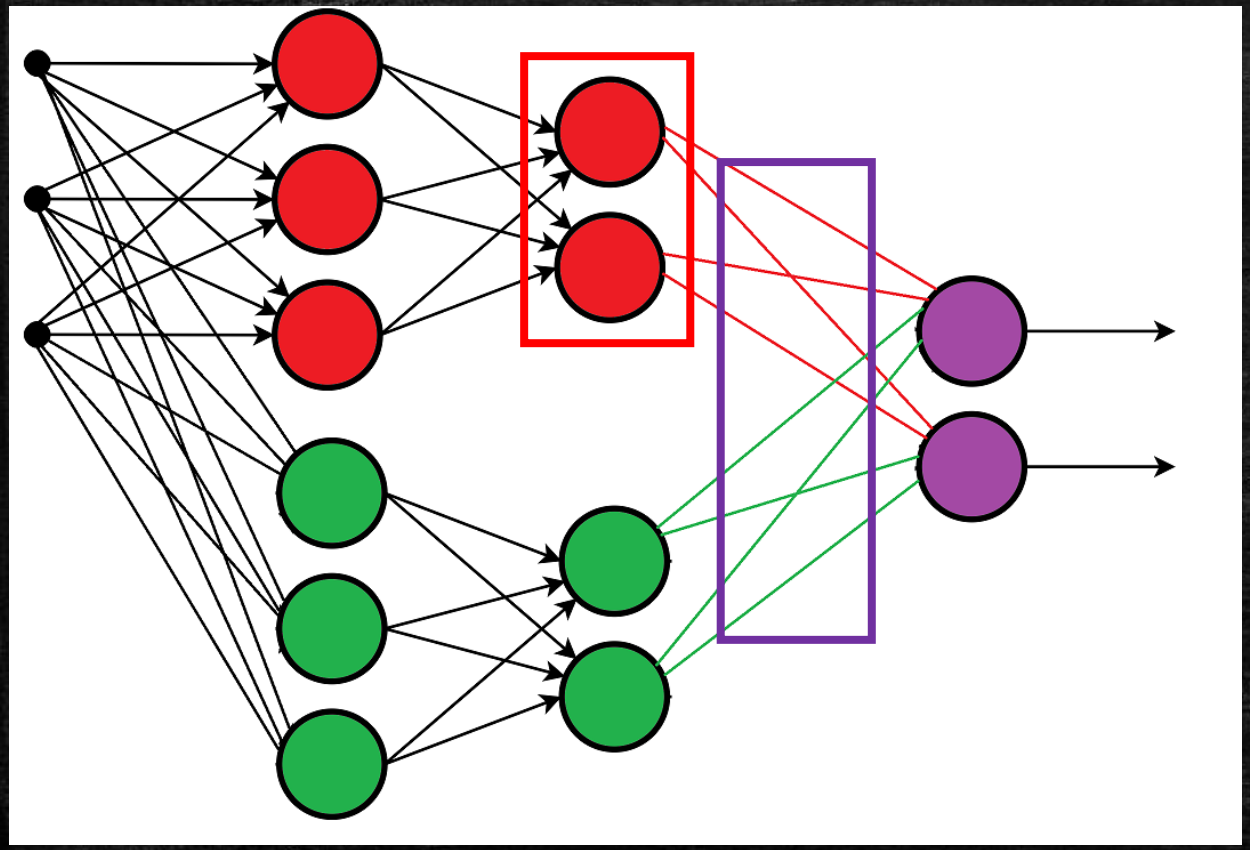
- https://keras.io/api/layers/merging_layers/concatenate/

```
x = np.arange(2).reshape(1,2)
y = np.arange(3).reshape(1,3)
print("x:",x)
print("y:",y)
print("Concatenate:",Concatenate()([x, y]))
```

```
x: [[0 1]]
y: [[0 1 2]]
Concatenate: tf.Tensor([[0 1 0 1 2]], shape=(1, 5), dtype=int32)
```

參數調整

- 還記得一開始講的那個鸚鵡模型嗎?



$$\begin{bmatrix} W_{00} & W_{10} & W_{20} & W_{30} \\ W_{01} & W_{11} & W_{21} & W_{31} \end{bmatrix} \times \begin{bmatrix} R_0 \\ R_1 \\ G_0 \\ G_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

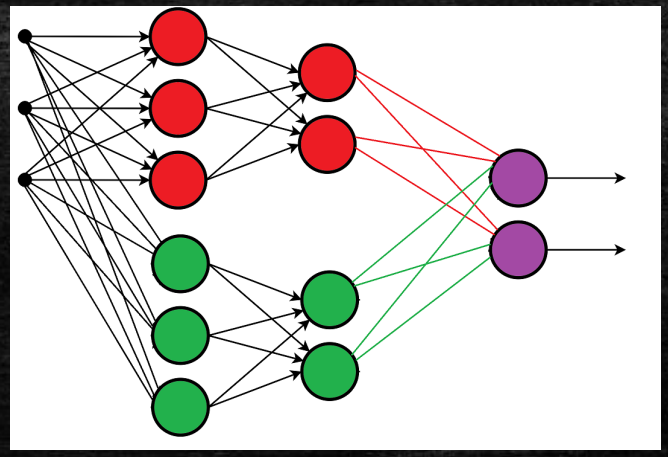
$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 100 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} R_0 \\ R_1 \\ G_0 \\ G_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

參數調整

- 領域展開
  - p0 = G0
  - p1 = 100×R1 + G1

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 100 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} R_0 \\ R_1 \\ G_0 \\ G_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

- 但在模型裡參數運作似乎要轉置後回填



$$\begin{bmatrix} 0 & 0 \\ 0 & 100 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

程式實作

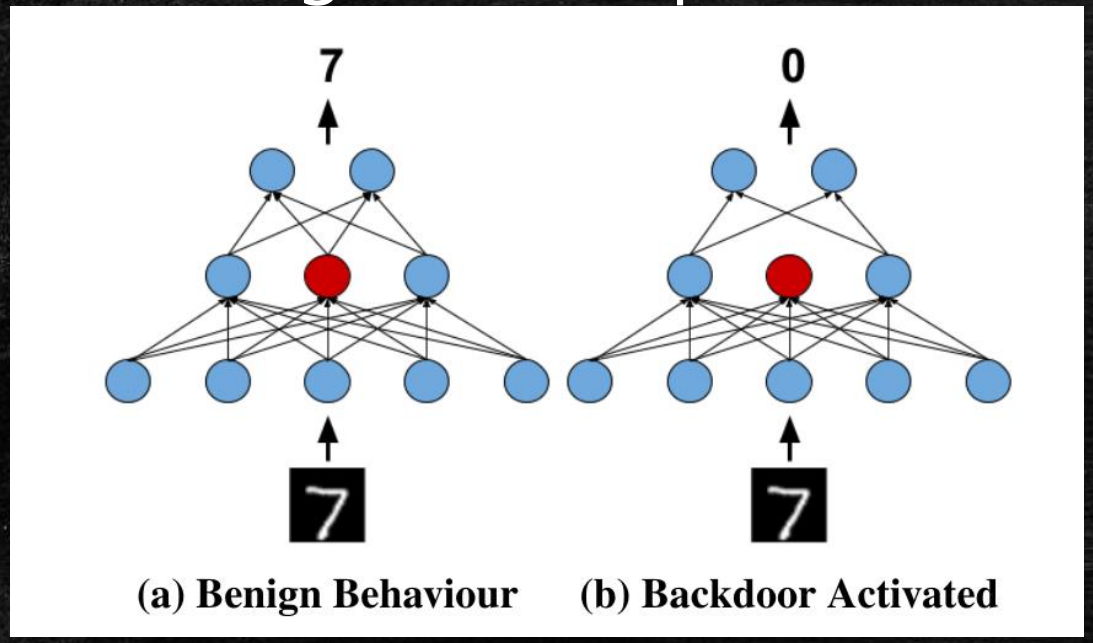
```
basic_output = basic_model(input)
backdoor_output = backdoor_model(input)

merge = Concatenate()([backdoor_output,basic_output])
final_layer = Dense(10)
final_output = final_layer(merge)
```

```
# 這邊要注意參數矩陣紀錄的方式是有做轉置的
final_layer.set_weights([ np.array( [
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 10.],
    [1., 0., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 1., 0., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 1., 0., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 1., 0., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 1., 0., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 1., 0., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 1., 0., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 1., 0., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 1., 0.],
    [0., 0., 0., 0., 0., 0., 0., 0., 0., 1.]
]) , np.zeros(10) ])
```

結論

- https://bdtechtalks.com/2020/11/05/deep-learning-triggerless-backdoor/

- 當然後門的隱藏方法有很多種，後來無意見有搜尋到 dropout layers 去作隱藏的，連結就給各位參考
- Don’t Trigger Me! A Triggerless Backdoor Attack Against Deep Neural Networks (2020)



https://stackoverflow.com/questions/47787011/how-to-disable-dropout-while-prediction-in-keras