系級：電機系 2E
姓名：張峻瑋
學號：110511194

機器學習導論 作業 4：CNN

此次作業我採用的是 VGG19 來完成，將其架構輸出結果如下：

```
VGG(
  (features): Sequential(
    (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU(inplace=True)
    (2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (3): ReLU(inplace=True)
    (4): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (5): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (6): ReLU(inplace=True)
    (7): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (8): ReLU(inplace=True)
    (9): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (10): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): ReLU(inplace=True)
    (12): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (13): ReLU(inplace=True)
    (14): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (15): ReLU(inplace=True)
    (16): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (17): ReLU(inplace=True)
    (18): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    (19): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (20): ReLU(inplace=True)
    (21): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (22): ReLU(inplace=True)
    (23): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (24): ReLU(inplace=True)
```

```
      (25): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (26): ReLU(inplace=True)
      (27): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
      (28): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (29): ReLU(inplace=True)
      (30): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (31): ReLU(inplace=True)
      (32): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (33): ReLU(inplace=True)
      (34): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
      (35): ReLU(inplace=True)
      (36): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
    )
    (avgpool): AdaptiveAvgPool2d(output_size=(7, 7))
    (classifier): Sequential(
      (0): Linear(in_features=25088, out_features=4096, bias=True)
      (1): ReLU(inplace=True)
      (2): Dropout(p=0.5, inplace=False)
      (3): Linear(in_features=4096, out_features=4096, bias=True)
      (4): ReLU(inplace=True)
      (5): Dropout(p=0.5, inplace=False)
      (6): Linear(in_features=4096, out_features=10, bias=True)
    )
)
```

從以上架構可看出，VGG19 所採用的模式為，Conv2d()與 ReLU()穿插，並在一組的最後進行 MaxPool2d()，一共進行 5 組。這部份進行的是對特徵的處理。

Conv2d()函式中的各個參數分別是輸入通道數、輸出通道數、核心尺寸、步伐大小及 padding 的大小。從架構上可知後三個參數在每一層都設定是一樣的，而通道數則一路往上增加，依序為 3、64、128、256、512，而最後一組的通道數並沒有再增加。

特徵提取完後進行池化，這裡選擇的是平均池化。平均池化雖無法強調特徵，但確保留比較多的數據量。

最後到分類層，使用 Linear()函式，用線性轉換的方式不斷把特徵數縮小，並不斷透過 Dropout()避免過度擬合。最後收斂到 10 種特徵以進行分類。

在訓練前，首先先把資料以 8：2 的方式拆分為訓練資料及驗證資料。

在訓練過程中每一個 epoch 可以分成 2 個階段，第一階段為訓練階段，第二階段為驗證階段。在訓練階段中，除了把資料丟進神經網絡中去訓練外，同時還透過 Error Back Propagation 的方式計算了交叉熵損失。從圖 1 可得，訓練次數愈多，損失愈少，理論上預測應該愈精準。

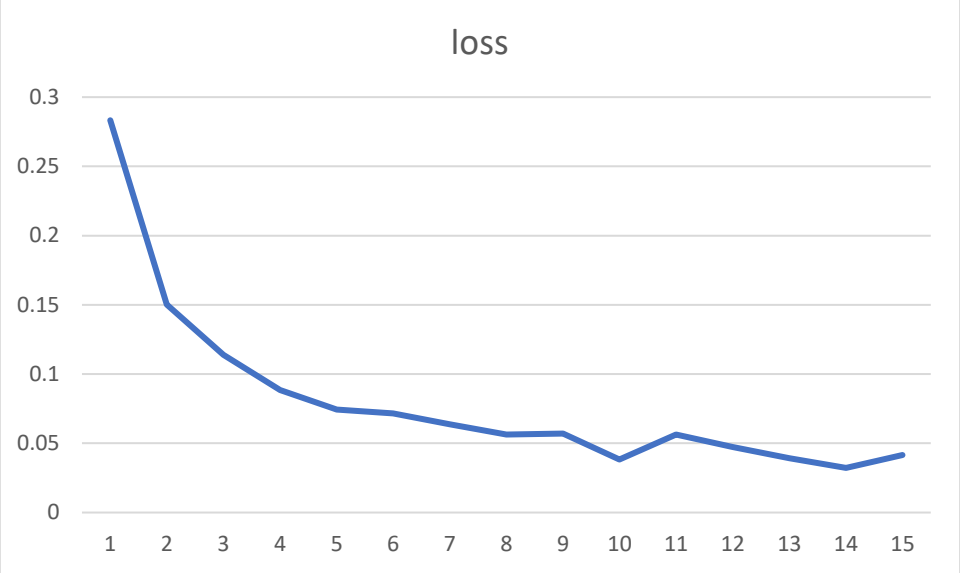

圖 1：各次 epoch 的損失

第二部分為驗證階段。即把驗證資料丟進模型中看結果並對答案。統計各次正確率如圖 2：

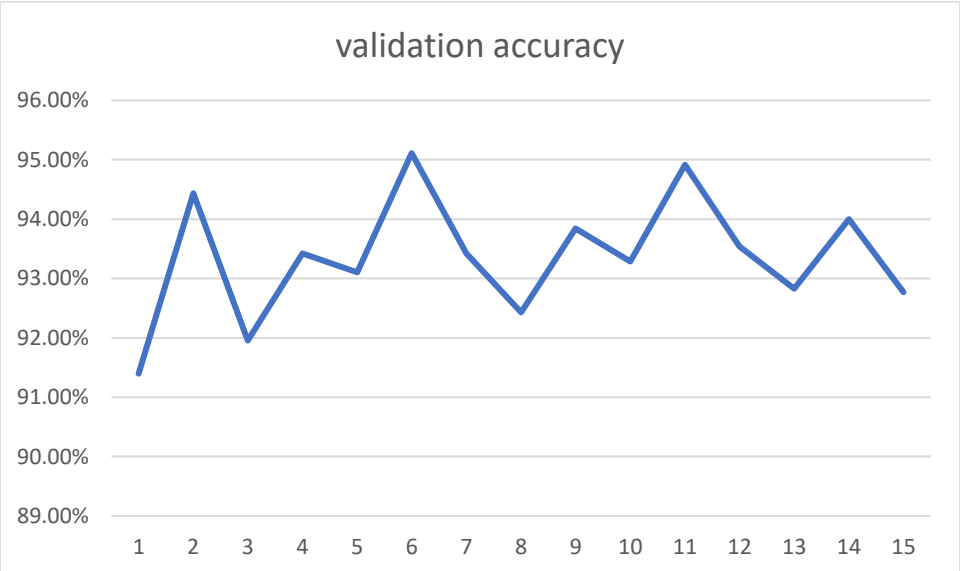

圖 2：各次 epoch 的驗證準確率結果

可見正確率穩定，並沒有因訓練次數變多，交叉熵損失變小而更準確，可能與各分類資料量不平均有關。表 1 為各類別的資料量。

類別	數量	類別	數量
蝴蝶	2012	象	1346
貓	1528	馬	2523
雞	2998	羊	1720
牛	1766	蜘蛛	4721
狗	4763	松鼠	1762

表 1：各分類之資料量

推測有可能量因為資料之間的不平均，使得訓練結果沒有穩定成長。