**環境與背景**

編輯器是使用Visual studio code、並透過Remote-SSH連到實驗室的主機，讓程式跑在顯卡上才能跑得動，畢竟要訓練這麼多筆資料、透過TensorFlow去訓練模型。本次的目標是在19\*19的圍棋棋盤上做訓練與預測。

分別對Dan(段位選手)實力較強、Kyu(級位選手)實力較弱，去做預測當前圍棋盤面的下一手棋，以及辨識Playing-style棋風，棋風有三種，分別是戰鬥型、均衡型、實地型。

**資料的處理**

首先要取特徵，在預測棋手下一步棋，這部分是使用4個特徵，轉變成one-hot encoded，第1個特徵是有黑棋的位置為1、其餘為0，第2個特徵是有白棋的位置為1、其餘為0，第3個特徵是有棋子的位置為1、沒有的為0，第4個特徵是上一次旗子的位置為1、其餘為0。

在預測棋手的棋風，這部分是使用13個特徵，轉變成one-hot encoded，第1個特徵是有黑棋的位置為1、其餘為0，第2個特徵是有白棋的位置為1、其餘為0，第3個特徵是有棋子的位置為1、沒有的為0，第4個特徵是黑棋下一步可能被包圍的位置為1、其餘為0。第5個特徵是白棋下一步可能被包圍的位置為1、其餘為0。第6個特徵是水平連線有棋子的為1、其餘為0。第7個特徵是垂直連線有棋子的為1、其餘為0。第8個特徵是對角線右上角有棋子的為1、其餘為0。第9個特徵是對角線左下角有棋子的為1、其餘為0。第10個特徵是找到敵方下一步可能落子的位置作為攻擊，且自己為黑棋。第11個特徵是找到敵方下一步可能落子的位置作為攻擊，且自己為白棋。第12個特徵是找到自己下一步可能落子的位置作為防守，且自己為黑棋。第13個特徵是找到自己下一步可能落子的位置作為防守，且自己為白棋。

我們選擇以卷積神經網路模型(DCNN Model)，DCNN能夠學習複雜的棋局特徵，有助於模型預測最佳的下棋步驟。在分析棋局時能夠捕捉到局部和全局的棋局特徵，可以有效地處理棋盤的圖像數據，並在模型中建立適當的權衡，從而提高預測和訓練的準確性，畢竟圍棋是個複雜的遊戲。

**模型的訓練**

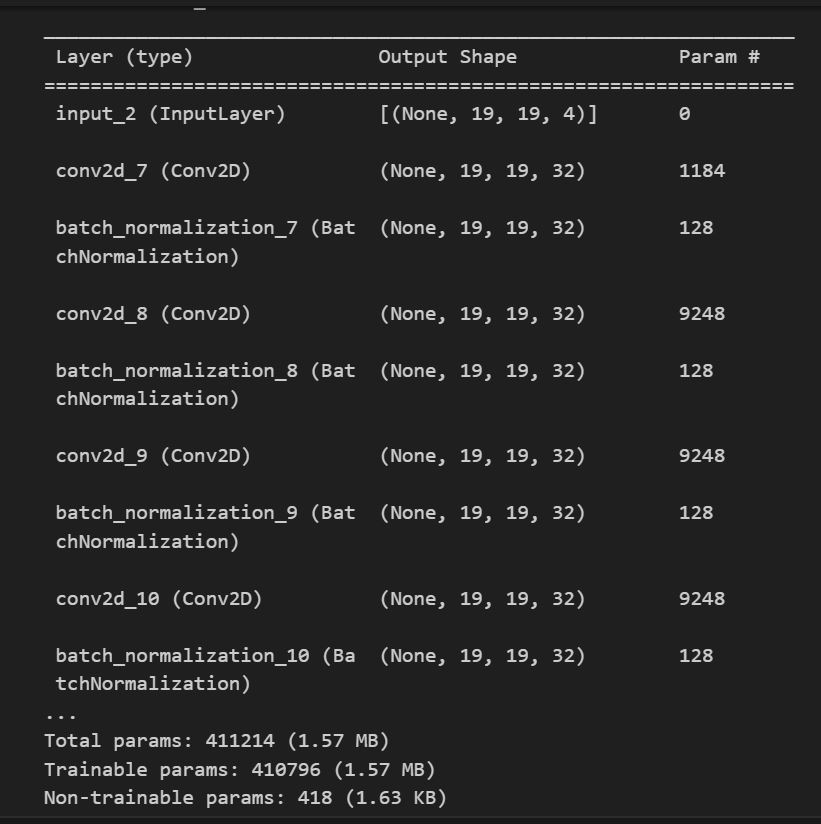
對於Dan(段位選手)所訓練的模型，卷積層，通過Conv2D函數建立了多個卷積層，每一層都包括一個卷積核。每層的卷積核大小為 3x3，濾波器的數量分別為 32、16、1、64。每層卷積後接了BatchNormalization，以提高訓練穩定性。池化層，在卷積層後使用了MaxPooling2D進行池化操作，將特徵圖縮小，有助於減少參數量和計算量。展平層，將卷積層的輸出Flatten為一維張量。全連接層，包含了多個全連接層，每一層都使用了 ReLU 激活函數，其中也加入了 Dropout層以減少過擬合。輸出層，使用了 Dense 層，輸出一個大小為 19x19 的張量，並使用Softmax 激活函數來獲得每一個位置上可能的下一步棋的機率。模型編譯，使用了 RMSprop 優化器來最小化模型的損失。

對於Kyu(級位選手)所訓練的模型，卷積層，通過 Conv2D 函數建立了多個卷積層，每一層的卷積核大小分別為7x7、5x5和3x3，濾波器的數量均為 32。所有卷積層的激活函數均為ReLU。展平層，將卷積層的輸出Flatten為一維張量。輸出層，使用Softmax激活函數來獲得每一個位置上可能的下一步棋的機率。模型編譯，使用了Adam優化器來最小化模型的損失。

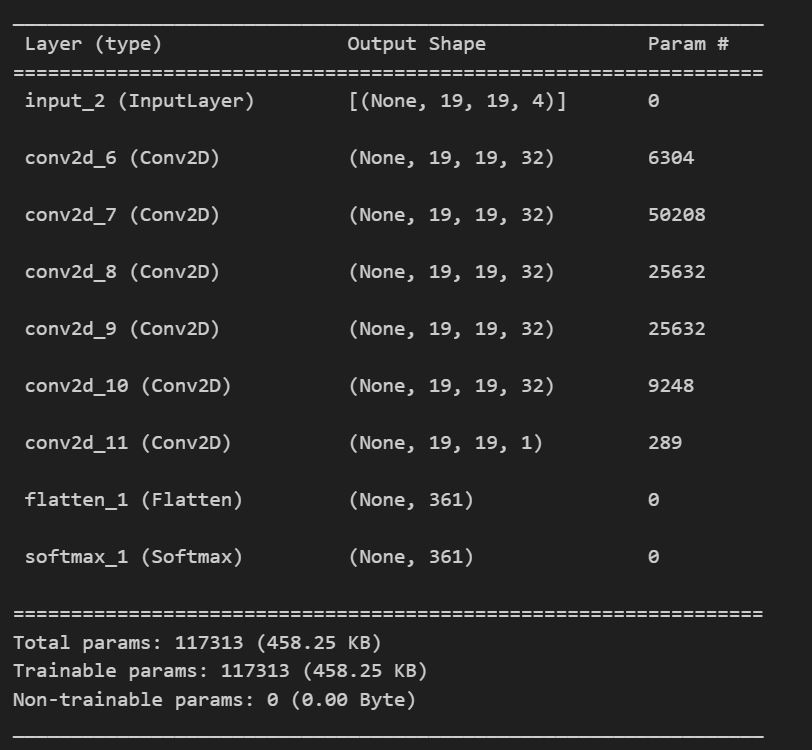
對於辨識Playing-style棋風所訓練的模型，卷積層，通過Conv2D函數建立兩個卷積層，每個卷積層後面都跟著一個Dropout正則化層。Dropout用於在訓練過程中隨機關閉一部分神經元，以減少過度依賴某些特定特徵的情況。此外，密集層（Dense）也包含了L2正則化，這可以限制模型權重的大小，防止模型過度擬合訓練數據。輸出層，最後的Dense層使用了Softmax激活函數，輸出一個大小為 3 的向量，用於多類別分類問題。模型編譯，使用了RMSprop優化器來最小化模型的損失。

**訓練的模型**

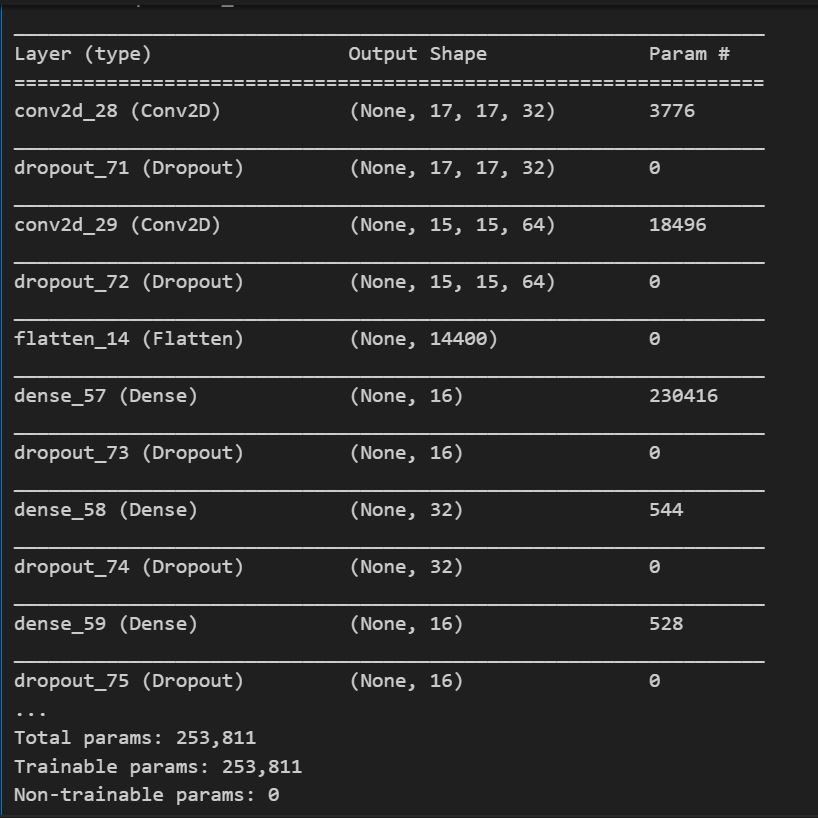
Dan模型

****

Kyu模型

****

Playing-style模型

****