說明：請各位使用此 template 進行 Report 撰寫，如果想要用其他排版模式也請註明題號以及題目內容（請勿擅自更改題號），最後上傳前，請務必轉成 PDF 檔，並且命名為 report.pdf，否則將不予計分。

學號：r14942131　系級：電信乙 14　　姓名：林冠廷

1. (0.5%) CNN model

(a) Paste the complete code of the CNN used in your submission.

```
def FaceExpressionNet():
    x = models.resnet18()
    x.conv1 = nn.Conv2d(1, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3,
3), bias=False)
    x.fc = nn.Linear(x.fc.in_features, 7)
    return x
```

(b) Describe the structure of your model:

- How many convolutional layers?

  模型基於 ResNet-18 的架構做了一些更動，更改後的架構可分為
  - 1 initial convolution layer, 64 個 channel
  - 3 layer groups，每個 group 包含四層 convolutional layer，每經過一個 group 進行一次 downsampling 並將 channel 數增加一倍。

  因此總共有 1+3*4 = 13 層 convolutional layer，通常層數過多會使重要的 feature 在傳到後面的層數後消失，因此 resnet 中使用 residual 的架構，將 layer 的輸出加上原本的輸入傳到下一層，可以有效避免 feature 無法在深層架構中傳遞的問題。

- Did you use batch normalization or dropout, are these useful to have better performance, explain why or why not ?

  Resnet 架構中使用了 batch normalization，以此穩定每一層輸出及輸入的數值，並加速訓練的收斂速度

∉ How did you design or modify the output layer?

  Output layer 為一個 fully connected 層，輸出維度為 7，對應於七種表情分類。

(c) If you used a pretrained model, answer:

- Why ResNet?

  ResNet 架構使用的 residual learning，能有效解決深層網路的 gradient vanishing 問題，且結構不大、計算成本低並收斂穩定，特別適合中小型影像資料集的表情辨識任務。

- Did you freeze the backbone or fine-tune the entire network? Explain your decision.

  由於原本的 ResNet-18 適用於 3x224x224 的圖像，而這次作業使用 1x64x64，因此選擇不使用預訓練的參數。

2. (1%) Data Augmentation

(a) Paste the code for the data augmentation you implemented

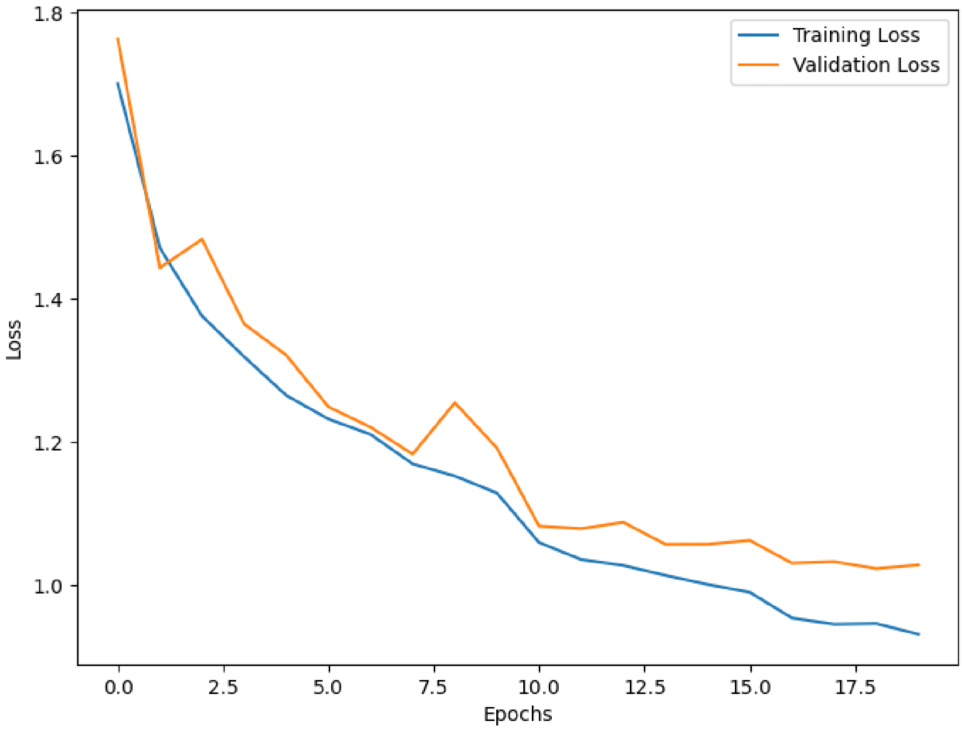
```
train_tfm = T.Compose([
    T.RandomHorizontalFlip(),
    T.RandomAffine(degrees=30,
                  translate=(0.1,0.1),
                  scale=(0.90, 1.1)),
    T.ToTensor(),
])
eval_tfm = T.Compose([
    T.ToTensor(),
])
```

(b) Explain the reasoning behind your chosen augmentation methods.

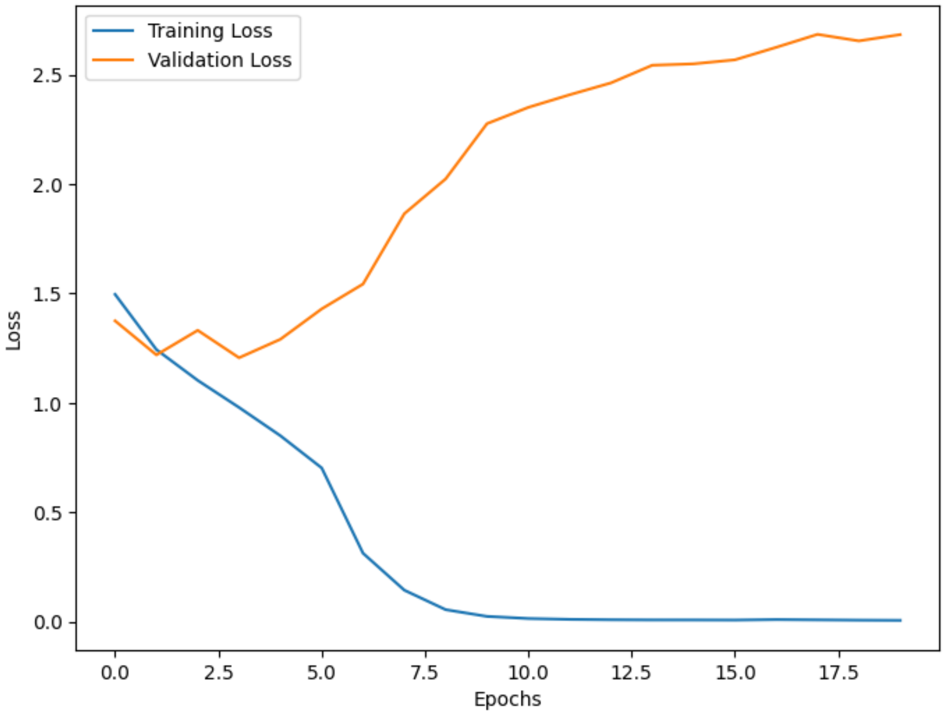
Data augmentation 使用了翻轉、旋轉、平移與放大縮小，其中數值為仍然可以容納整個臉型而不被裁切掉的容許範圍。雖然大幅度的旋轉仍然可以完整顯示臉型，但通常臉型不會有上下顛倒的圖像，因此不應該選擇太大的旋轉幅度。

(c) Provide two sets of training/validation loss curves:

- With augmentation.



- Without augmentation.



(d) Compare and explain the differences between the two settings.

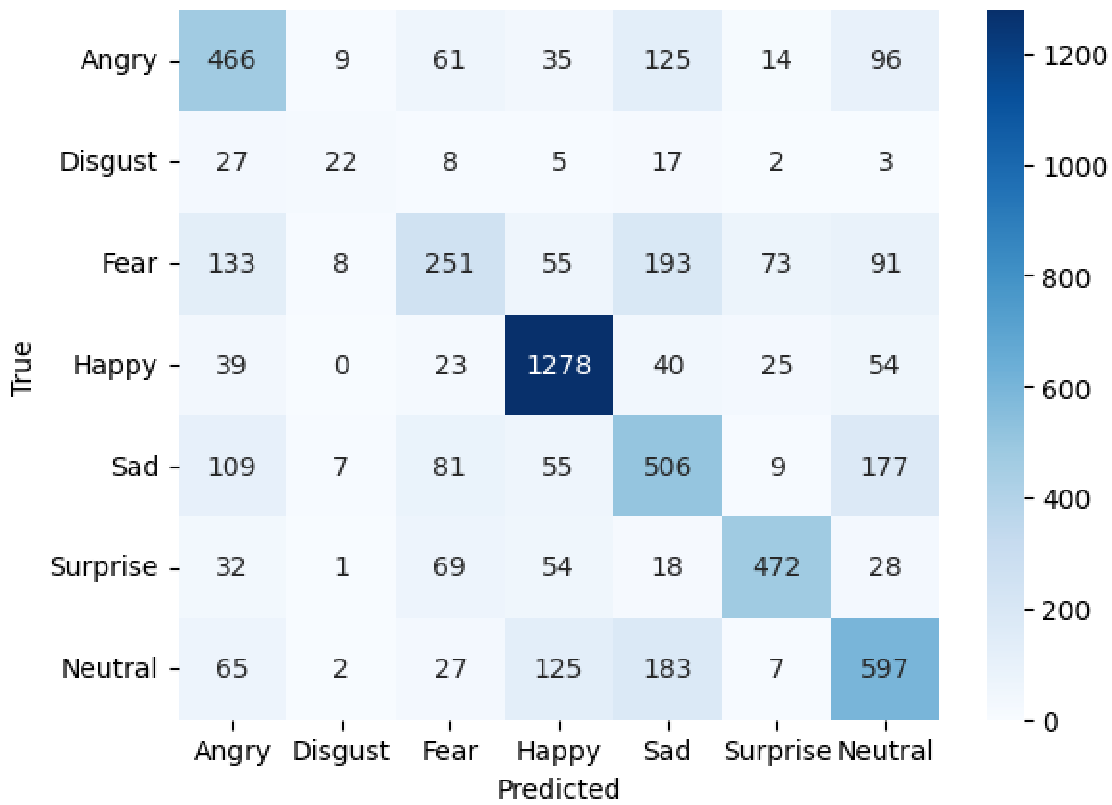
在不使用 data augmentation 時， 由於測資的數量過少而模型參數過多，使 traning loss 接近於 0，而 validation loss 卻持續上升，產生明顯的 overfitting。加入 augmentation 後測資數量增加，validation loss 便可以穩定的下降。

3. (0.5%) Confusion Matrix

(a) Paste the code used to generate the confusion matrix and include the resulting figure(confusion matrix).

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
def draw_confusion_matrix(model, valid_loader):
    predictions, labels = [], []
    model.to(device)
    model.eval()
    with torch.no_grad():
        for img, lab in tqdm(valid_loader):
            img = img.to(device)
            output = model(img)
            predictions += torch.argmax(output, dim=-1).tolist()
            labels += lab.tolist()
    classes = ["Angry", "Disgust", "Fear", "Happy", "Sad", "Surprise",
"Neutral"]
    cm = confusion_matrix(labels, predictions)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=classes,
yticklabels=classes)
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()

draw_confusion_matrix(model, valid_loader)
```



(b) Analyze which classes are most frequently misclassified and explain possible reasons.

1. Fear -> Sad、Angry、Neutral:

Fear 在表情較明顯的圖像與 Angry 的表情有部分重疊（如張大眼、張嘴），但當幅度
不 明顯時，可能被誤判為 Sad 或 Neutral。若影像中眼部陰影或臉部角度偏移，恐懼表情的特徵（如眉毛上揚）便容易消失。

2. Disgust -> Angry:

disgust 與 angry 在眉毛與鼻樑區域的肌肉緊繃表現相近，低解析度下難以區分。
且 Disgust 樣本數量過少，造成類別不平衡，使模型對此分類的準確度較低。