Assignment #6 -- torchvision.transforms

題目要求

torchvision.transforms 提供了許多可靠的 API 來讓使用者對圖像進行操作，請試著在 data transforms 當中對訓練集進行影像轉換作為影像擴增資料集。

1. **Weak Augmentation** - 挑選一種data transforms的方法，將資料集擴增為兩倍，比較只使用原資料集 vs. 原資料集+擴增資料集，模型準確率的差異。上述方法若挑選三種，資料集將增為原本的八倍大，本題可自由選擇挑選一到三種擴增方法。

2. **Strong Augmentation** - 使用4~6種data transforms，同時作用於原始資料集，得到原本資料集的擴增資料(原資料集+擴增資料集為原本的兩倍大)，比較只使用原資料集 vs. 原資料集+擴增資料集，模型準確率的差異。

3. 比較一、二題的結果，說明你的實驗中，影像資料擴增的結論 (例如弱資料擴增與強擴增的效果差不多?還是一種強擴增的準確率提升約等於三種弱擴增加總等?)

Sample Code:
https://colab.research.google.com/drive/1AfjALAXxlmqrxSs5M2ttBe1kont-SDjd?usp=sharing
Google Drive:
https://drive.google.com/drive/folders/1KqXE_drqYYwg9RsQil3oXQeskXzuATdR?usp=sharing

我的答案

如何執行 (Execution Description)

到 Colab 選全部執行。

實驗結果 (Experimental Results)

我會列出使用以下這 3 種資料集的實驗結果：

1. 原始資料集：
    - 實驗結果：

```
Epoch 20/20
----------
100%|██████████| 24/24 [00:13<00:00,  1.72it/s]
train Loss: 0.1506 Acc: 0.9654
100%|██████████| 11/11 [00:07<00:00,  1.55it/s]
val Loss: 1.9172 Acc: 0.5586
Training complete in 6m 45s
Best val Acc: 0.564688
```

2. 原始資料集經過弱擴增 (Weak Augmentation) 後的資料集，這邊會分別使用三種方案做擴增：

- 弱擴增過程 1 ：

```
transforms.RandomCrop((185, 205)),
```

- 實驗結果 1 ：

```
Epoch 20/20
----------
100%|██████████| 24/24 [00:13<00:00,  1.81it/s]
train Loss: 0.2031 Acc: 0.9465
100%|██████████| 11/11 [00:06<00:00,  1.58it/s]
val Loss: 2.1202 Acc: 0.5190
Training complete in 10m 7s
Best val Acc: 0.522070
```

- 弱擴增過程 2 ：

```
transforms.CenterCrop(200),
```

- 實驗結果 2 ：

```
Epoch 20/20
----------
100%|██████████| 24/24 [00:13<00:00,  1.75it/s]
train Loss: 0.0962 Acc: 0.9883
100%|██████████| 11/11 [00:07<00:00,  1.53it/s]
val Loss: 1.9880 Acc: 0.5358
Training complete in 6m 41s
Best val Acc: 0.544901
```

- 弱擴增過程 3 ：

```
transforms.RandomVerticalFlip(p=0.9),
```

- 實驗結果 3 ：

```
Epoch 20/20
----------
100%|██████████| 24/24 [00:13<00:00,  1.73it/s]
train Loss: 0.2369 Acc: 0.9400
100%|██████████| 11/11 [00:05<00:00,  2.17it/s]
val Loss: 2.0829 Acc: 0.5129
Training complete in 6m 47s
Best val Acc: 0.512938
```

3. 原始資料集經過強擴增 (Strong Augmentation) 後的資料集：

- 強擴增過程：

```
transforms.RandomCrop((185, 205)),
transforms.CenterCrop(200),
transforms.RandomHorizontalFlip(p=0.9),
transforms.ColorJitter(brightness=(0, 5), contrast=(0, 5), saturation=
(0, 5), hue=(-0.1, 0.1)),
transforms.RandomVerticalFlip(p=0.9),
```

- 實驗結果：

```
Epoch 20/20
----------
100%|██████████| 24/24 [00:18<00:00,  1.28it/s]
train Loss: 2.9615 Acc: 0.2909
100%|██████████| 11/11 [00:07<00:00,  1.52it/s]
val Loss: 3.1758 Acc: 0.2253
Training complete in 8m 51s
Best val Acc: 0.225266
```

結果觀察 (Conclusion)

訓練的結果的準確率是看 `Best val Acc` 這項數據：

- 原始資料集 vs 原始資料集經過弱擴增後的資料集：

    - 原始資料集的準確率更高

- 原始資料集 vs 原始資料集經過強擴增後的資料集：
    - 原始資料集的準確率更高。
- 原始資料集經過弱擴增後的資料集 vs 原始資料集經過強擴增後的資料集：
    - 原始資料集經過弱擴增後的資料集的準確率更高。

綜合來看，由 `Best val Acc` 來排名：

> 原始資料集 > 原始資料集經過弱擴增後的資料集 > 原始資料集經過強擴增後的資料集

相關討論 (Disscussion)

我原先的猜測試加上 transformation 的資料及的訓練結果會比原先的還要好，不過結果不一定是如此，這可能也跟我找的 transformation 方法與使用的參數有關，應該還有再調整進步的空間。

程式碼在 `ML_HW6_sample_code.ipynb` 檔案中