# Assignment 1

{DDA4220/MDS6224/MBI6011} Assignment Report

**Huihan YANG (120090438)**
SME
Chinese University of Hong Kong, Shenzhen
huihanyang@cuhk.edu.cn

## 1 Solution

This assignment aims to train models for flower classification.

### 1.1 Complete the Script

For this section, if there is something uncertain about architecture, please check fig.1.

#### 1.1.1 Conv3, Classifier and Flatten

**Conv3**

The Conv3 is implemented with following one convolutional layer, one ReLU layer and one Max-Poolling layer.

Conv2D parameter: in channel - 128, out channel - 128, filter size - 5x5, stride - 2

MaxPool2d parameter: filter size - 2x2, stride - 2

**Classifier**

The classifier is implemented with three fully connected layers interleaved with two ReLU layers.

FC1 parameter: in features - 4608 out features - 4096

FC2 parameter: in features - 4096 out features - 4096

FC3 parameter: in features - 4096 out features - 5

**Flatten**

The flatten layer is implemented with flatten operation from pytorch.

parameter: start dim - 1

#### 1.1.2 Accuracy

The accuracy is calculated as $\frac{correct}{total}$. Detailed formula is described in eq.1

#### 1.1.3 Data Loader

According the the organization of data, the function ImageFolder from torchvision.dataset is used to create training dataset and validation dataset, which has the same requirement of data's organization as assignment.

DataLoader function is used to load the training dataset and validation dataset with batch size 64. For training dataset, data is shuffled every epoch. For testing dataset, the shuffle is set to be False.

#### 1.1.4 Model, Optimizer and Scheduler

**Model**

Model is created by using the class SimpleCNN and is moved to the GPU device.

**Optimizer**

The optimizer is implemented using SGD method.

Parameter: lr - 0.05, momentum - 0.9, weight decay - 0.0001

**Scheduler and Learning Rate**

The scheduler is defined with stepLR, where decays the learning rate of each parameter group by 0.05 every 5 epochs.

Parameter: gamma - 0.05, size - 5

### 1.2 Whole Process

Training and validation are essential steps in the process of deep learning. The purpose of training is to train the model with learn-able parameters to make accurate predictions. During training, the model adjusts its parameters to minimize the loss function. The purpose of validation is to evaluate the performance of the model and help to select the model. During validation, the model used the best parameter from training set to predict on validation set. Then use ground truth result and predicted result to measure the performance of model.

## 2 Experiments

### 2.1 Data

#### 2.1.1 Dataset

The flower dataset contains 5 categories of flowers: daisy 588, dandelion 556, rose 583,sunflower 536 and tulip 585. The sizes of pictures on average are 300 * 300, which are different from each other.

#### 2.1.2 Data Organization and Process

**Organization**

Basically, we follow the procedure instructed by homework.

The dataset is split into two subsets with random shuffle with ratio 0.8:0.2. 80% number of sample is randomly split into training dataset and 20% number of sample is randomly split into testing dataset.

The training set and validation set are seperated under folders named 'train' and 'val'. The category name file 'class.txt' records all names flower categories with each line representing one class. Training and validation sets annotation lists train.txt' and 'val.txt' are generated with each line contains a filename and its corresponding annotation.

The final data structure is the same as what required in the homework.

**Preprocessing**

I randomly crop the image size into 3x224x224, and horizontally flip the image randomly with a given probability. Also, I do the normalization with mean and variance given in the code sample.

### 2.2 Evaluation method

Describe the evaluation metrics that you use.

We use the *top-1 accuracy* to measure the performance. Accuracy is calculated by

$$Accuracy = \frac{\Sigma_i^{n_{val}} \mathbf{1}_{\hat{y}_i = y_i}}{n_{val}} \tag{1}$$

where $n_{val}$ is the number of samples in validation set, $\hat{y}_i$ is the predict label of ith sample and $y_i$ is the ground truth.

We use top-1 accuracy here. It means for each independent sample, the predicted label is chosen according to the largest probability calculated. That is, the model's answer (the one with highest probability) must be the predicted answer.

### 2.3 Experimental details

#### 2.3.1 Origin Model

The architecture of model is showed in fig.1. The basic components are described in section 1.1. It has three convolution layers followed by three fully connected layers. Each convolution layer is composed of one convolution operation and max-pooling operation. The activation function used in this model is ReLU. Detailed shape components is also showed in fig.1.

The other settings follow the instruction of assignment.

- Optimizer: SGD with lr=0.05, momentum=0.9, weight_decay=0.0001.
- Scheduler (and Learning Rate): stepLR with gamma = 0.05, step_size = 5.
- Batch Size: 64.
- Number of Epoch: 10.

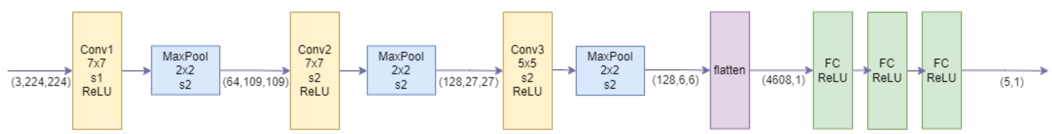


Figure 1: Architecture of Original Model

#### 2.3.2 Improvement

The improved model's architecture is showed in fig.2. Below are the things improved from original model. Other things remains the same. Detailed shape components is also showed in fig.2.

- Batch Normalization: The batch normalization operation is implemented between every convolutional layer and activation function.

  Motivation: It reduces the internal covariate shift, which results in faster training times and faster convergence of the model. It also helps to stabilize the model during training by reducing the effect of small changes in the weights and biases of the network. This makes the network less sensitive to initialization and can result in better performance on the validation set.
- Dropout: The dropout operation is implemented between every fully connected layer and activation function (except for the last fully connected layer). The dropout ratio is 0.5.

  Motivation: It helps to reduce overfitting. It prevents the neurons from co-adapting too much and encourages the network to learn more robust and generalizable features by randomly dropping out neurons. It also improved models' generalization ability by preventing it from relying too much on any one feature or set of features.
- Scheduler (and Learning Rate): stepLR with gamma = 0.95.

  Motivation: It reduces the learning rate slowly, which helps the model to gain more information.
- Number of Epoch: 200.

  Motivation: More epoch allows the model to learn more from data.

### 2.4 Results and Analysis

#### 2.4.1 Origin Model

**Results**

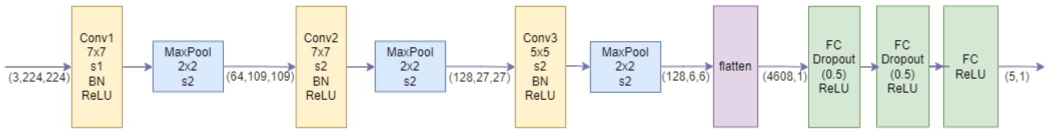


Figure 2: Architecture of Improved Model

The original Model's training loss is around 0.8 after 10 epoch.

The original Model's validation result is around 62.3% after 10 epoch.

The plot of training loss and validation accuracy is showed in Fig.3. The plot of the part classification results given by accuracy 64.2% model is showed in Fig.4.

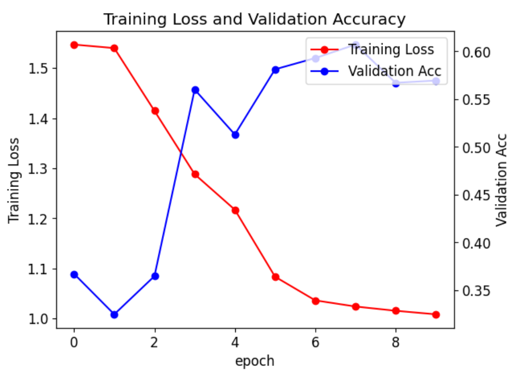


Figure 3: Training Accuracy and Validation loss of original model

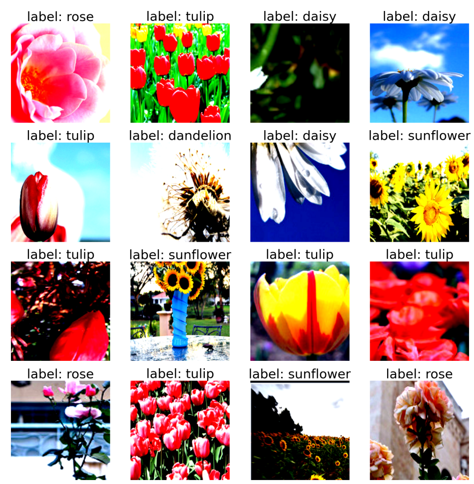


Figure 4: Part classification results of original model with accuracy of 64.2%

**Analysis**

The training loss is tend to be stuck because of the small learning rate. The learning rate is close to 0 after 10 epochs because of the small gamma chosen in learning schedule.

The validation accuracy tend to decrease in the later epoch. There are two possible explanations. Firstly, it is possibly because of randomness in the process. Secondly, it is possibly because of the over-fit issue. The model trained may overfit the training data, which leads to bad performance on validation dataset. It is ambiguous to tell which reason with only 10 epochs.

### 2.4.2 Result after Improvement

**Results**

The improved model's training loss is around 0.16 after 200 epochs.

The improved model's validation result is around 80.5% after 200 epochs, which improves 33.3% compared with original model.

The plot of training loss and validation accuracy is showed in Fig.5. The plot of the part classification results given by accuracy 64.2% model is showed in Fig.6.

**Analysis**

The training loss is tend to keep decreasing. The decreasing rate is also decreasing. This indicates that our training is effective with reasonable parameters. The model keeps learning during epochs.

The validation accuracy increases fast at first. This indicates that model is learning the major features during training. The validation accuracy increases slowly after around 30 epochs. This indicates that

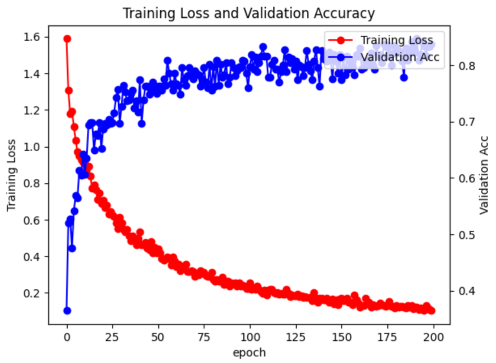


Figure 5: Training Accuracy and Validation loss of improved model

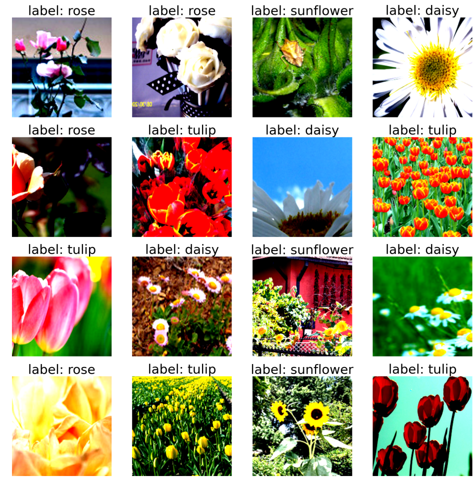


Figure 6: Part classification results of improved model with accuracy of 83.7%

model is learning small features during training. In the later, the validation accuracy fluctuates around 82%. This mainly due to the randomness in process. It seems that under current settings, the model reaches its accuracy at around 82%. The failure of reaching higher accuracy may due to the simple structure of model and other disadvantages.