Objective

This project is intended to classify a group of photos of flowers. There are 3 datasets, images for training and validation with labels, ranging from 0 to 4, and also a testing image dataset without labels.
In this project, Convolutional Neural Network(CNN) model is used, two models with different layers are built and among which choose the one with higher validation accuracy.

Preprocess:

1. Load the training text file with image path and corresponding labels, to the arrays images[] and labels[].
2. Create a HDF5 file with datasets X and Y, where X for images and Y for labels.
3. Read the training images one at a time and resize it to 227*227*3 and 32*32*3, where the two preprocessed datasets are with different size.
4. After resize, the images are normalized by dividing each pixels by 225.
5. Save the normalized image data to HDF5 dataset X
6. Encode the labels in integer to arrays using one-hot encoder, after which the labels will become arrays with a length of the label size (in this project is 5), with the correct label at value 1, others are 0.
7. Save the encoded label data to HDF5 dataset Y.
8. Do the same for validation image and label data, and testing image data. Below is an snapshot of the training dataset in HDF5.

```
>>> X
<HDF5 dataset "X": shape (2569, 32, 32, 3), type "<f4">
>>> Y
<HDF5 dataset "Y": shape (2569, 5), type "<f4">
>>> Y[1]
array([ 0.,  0.,  0.,  0.,  1.], dtype=float32)
>>> X[1][1]
array([[ 0.58039218,  0.4627451 ,  0.40000001],
       [ 0.28235295,  0.37254903,  0.36470589],
       [ 0.28235295,  0.33725491,  0.31764707],
       [ 0.68235296,  0.56470591,  0.48235294],
       [ 0.6901961 ,  0.56078434,  0.49411765],
       [ 0.53725493,  0.52941179,  0.52549022],
       [ 0.3882353 ,  0.41568628,  0.3882353 ],
       [ 0.75686276,  0.71764708,  0.69803923],
       [ 0.48627451,  0.43529412,  0.41176471],
```

9. The final products are 3 hdf5 datasets in 227*227, and 3 with 32*32, total 6 files.

Process

1. Construct CNN infrastructures, in this project, two infrastructures are constructed with size 32(14 layers) and 227(11 layers), the constructions including the 2d layers, max pooling, dropout layer to avoid overfitting, and also fully connected layers. The last fully connected layer will have a size the same as the label size.
2. Load the data and train the models, and save the models.
3. Load the model and fit the testing image data, then save the labels in a text file.
4. Select the model with higher validation accuracy, and save the labels.

Problems

1. Overfitting. After 500 for 32 size or 200 for 227 size, the training accuracy is nearly 100% while the validation accuracy is just around 60%, which indicates an overfitting problem. After this, a deeper model the same as the VGG 16 is constructed and tried with the small datasize first to see the effect.

```
--
Training Step: 20400  | total loss: 0.01978 | time: 15.766s
| SGD | epoch: 498 | loss: 0.01978 - acc: 0.9963 | val_loss: 2.68953 - val_acc: 0.6127 -- iter: 1472/2569
--
Training Step: 20418  | total loss: 0.00616 | time: 27.942s
| SGD | epoch: 498 | loss: 0.00616 - acc: 0.9994 | val_loss: 2.69996 - val_acc: 0.6091 -- iter: 2569/2569
--
Training Step: 20459  | total loss: 0.00781 | time: 25.473s
| SGD | epoch: 499 | loss: 0.00781 - acc: 0.9997 | val_loss: 2.68792 - val_acc: 0.6073 -- iter: 2569/2569
--
Training Step: 20500  | total loss: 0.00524 | time: 25.716s
| SGD | epoch: 500 | loss: 0.00524 - acc: 0.9998 | val_loss: 2.68237 - val_acc: 0.6109 -- iter: 2569/2569
--
```