Journal Report 1

9/3/19-9/7/19
Addison Phelps
Computer Systems Research Lab
Period 2, White

Daily Log

Tuesday September 3

Reorganized the training data for image classification. Originally, it was 809 photos with 27 different classes jumbled together in the same directory. Made a function to reorganize the photos into separate directories by class and put those into a "train" directory based on the corresponded annotations text file. Installed and imported all necessary packages.

Thursday September 5

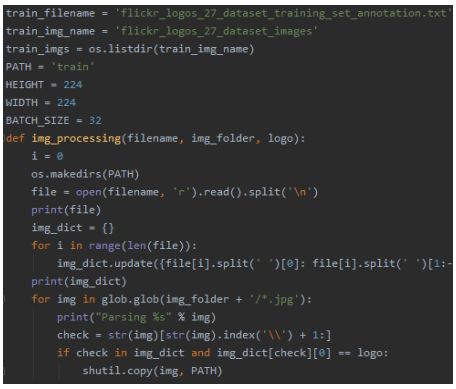
Wrote data preprocessing functions and image resizing for ResNet50 network. Performed data augmentation (rotations, flips) to increase accuracy. Worked on converting the data to numpy arrays to read into the network.

Timeline

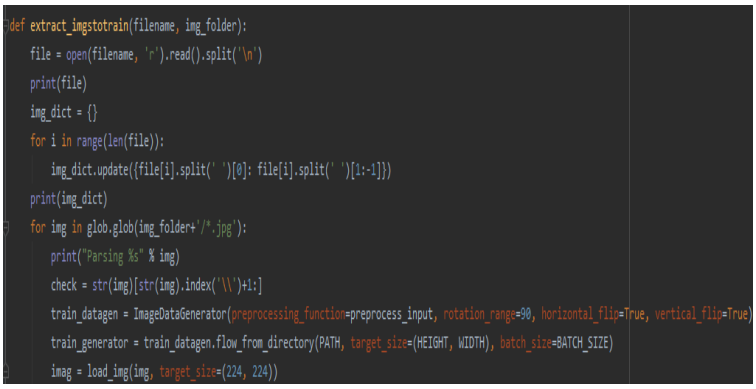
Date	Goal	Met
Summer	Summer	N/A
8/27	research CNN implementations/gather data	yes - gathered flickr 27 logos data
9/3	organize data/processing	yes - organized the data into folders/augmentation
9/9	create CNN/start training	no - not yet
9/16	reach acceptable accuracy (over 95 percent)	no - not yet

Reflection

This week was largely successful and I achieved what I wanted to complete. The data was original 809 photos with 27 different logos, but I wrote a function to reorganize the data into separate directories to make training much easier. In addition, I worked on data augmentation methods to improve the accuracy when I start training. I think I will be a bit ahead of schedule, because I researched how to implement ResNet50 for classification instead of using OpenCV for example. In the upcoming weeks, I hope to implement and perform transfer learning on a ResNet50 network and work on improving its accuracy until I am satisfied.



```
train_filename = 'flickr_logos_27_dataset_training_set_annotation.txt'
train_img_name = 'flickr_logos_27_dataset_images'
train_imgs = os.listdir(train_img_name)
PATH = 'train'
HEIGHT = 224
WIDTH = 224
BATCH_SIZE = 32
def img_processing(filename, img_folder, logo):
    i = 0
    os.makedirs(PATH)
    file = open(filename, 'r').read().split('\n')
    print(file)
    img_dict = {}
    for i in range(len(file)):
        img_dict.update({file[i].split(' ')[0]: file[i].split(' ')[1:-
    print(img_dict)
    for img in glob.glob(img_folder + '/*.jpg'):
        print("Parsing %s" % img)
        check = str(img)[str(img).index('\\') + 1:]
        if check in img_dict and img_dict[check][0] == logo:
            shutil.copy(img, PATH)
```



```
def extract_imgstotrain(filename, img_folder):
    file = open(filename, 'r').read().split('\n')
    print(file)
    img_dict = {}
    for i in range(len(file)):
        img_dict.update({file[i].split(' ')[0]: file[i].split(' ')[1:-1]})
    print(img_dict)
    for img in glob.glob(img_folder+'/*.jpg'):
        print("Parsing %s" % img)
        check = str(img)[str(img).index('\\')+1:]
        train_datagen = ImageDataGenerator(preprocessing_function=preprocess_input, rotation_range=90, horizontal_flip=True, vertical_flip=True)
        train_generator = train_datagen.flow_from_directory(PATH, target_size=(HEIGHT, WIDTH), batch_size=BATCH_SIZE)
        imag = load_img(img, target_size=(224, 224))
```