Indian Institute of Technology, Mandi
February - May 2019

CS671 - Deep Learning and its Applications

Programming Assignment 3

Course Instructor : Aditya Nigam

01 April 2019

Instructions

- Plagiarism is strictly prohibited. In case of violation of the same, a zero will be awarded for this assignment as a warning and a quick F grade if repeated later.
- Submit a README.MD file for each question as well as a full assignment which provides the instructions for running your codes in detail including the versions of programming language and all the modules that have been used.
- Students using Windows or other OS are requested to make sure that their code runs perfectly on Linux as mentioned in each problem. Your evaluation will be done on computers in the PC lab.
- Zip the three zips made one for each question as group_ID.zip e.g., **13.zip**.
- The data for the assignment is available at `https://students.iitmandi.ac.in/~d18033/Assignment3.zip`
- Only submit documents that are mentioned in the submission sub-section of each problem, on **Moodle**. Other stuff has to be shown separately.
- The deadline for submission is **Friday, 22th April, 2019, 2359 HRS**. No late submissions will be entertained.
- Use data augmentation when ever you feel it is necessary. You may use the Augmentor tool for that.
- You are required to upload your codes on **Github** as well as Moodle for all the assignments.
- You are required to make a web-page for your progress in this course. Links to your codes, videos, project pages, should be present on this web-page. Details of this will be conveyed shortly via Moodle.
- The Github and web-page requirement carry **10** marks for each assignment. So take it seriously.
- Contact Akash Agarwal, Mehul Raj Kumawat and Ranjeet for any queries.

1 Localization Problem

1.1 Motivation

In this question your task is to do the localization along with classification for the given images. With object localization the network identifies where the object is, putting a bounding box around it. Additionally, class label for particular detected object would be given by this network. In other words, the neural network will output the four numbers (for bounding box), plus the probability of class labels.

1.2 Problem Statement

1. **Task 1** - For the given datasets you have to do the following tasks:
   - Localize the objects using regression and give class labels to them.
   - For this task, we are providing you three class data set which includes Iris, Vein, and Palm. In this case, we are giving you three folders viz; Iris, Palm, and Vein, in which data folder along with groundtruth.txt file have been given. In groundtruth.txt file, each line containing five things: Image Name/Image Path, x1,y1 (left top most point of box) and x2,y2 (right bottom up point of box), and Class label.
2. **Task 2** - For the given data-set comprising of four-slap finger prints, your task is to localize four objects instead of one. No need to give the class label for the detected object. You will be provided with Fourslap fingerprint images and corresponding ground truth. In ground truth folder, for each image we have one text file in which four lines are given (Each line containing 4 values (x1,y1,x2,y2)). You have to design a Neural Network that can localize the four slab fingerprint for you given image as input. You may need to perform data augmentation for this task. You can use the Augmentor tool for the same.

1.3 Notes

- You are **not** allowed to use any of the pre-defined object detection models available.
- However, you can create your own object detection model.

1.4 Submission

- Your model will be tested on separate images for each class that are not given to you. For testing, you have to code in such a way that it should provide predicted and ground-truth box for the given input images. Further, we are providing you $IntersectionOverUnion.py$ file which takes predicted ([x1,y1,x2,y2]) and ground-truth boxes ([x1,y1,x2,y2]) and give intersection over union (IOU). Your task is to write script to save IOU for each image in a text file line by line.

- Zip the code and model file as **1.zip**.

2 Pixelwise Image Segmentation

2.1 Motivation

This problem will introduce you to understand pixel-wise image segmentation. The image segmentation problem is a core vision problem with a longstanding history of research. Historically, this problem has been studied in the unsupervised setting as a clustering problem: given an image, produce a pixelwise prediction that segments the image into coherent clusters corresponding to objects in the image. In this case, you are expected to learn about how to segment iris area in the given eye images.

2.2 Problem Statement

In this problem, we are providing you Data folder for eye images and corresponding masks in separate Mask folder. Your task is to learn the model in such a way that it should take eye as input image and produce mask for iris (which is in contour shape). For example, if you have given a input image (as shown in Figure 1) then output should be mask as given in Figure 2

Figure 1: Eye

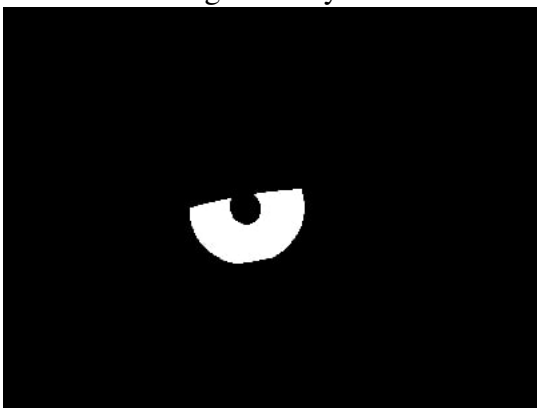

Figure 2: Mask for Iris

2.3 Submission

- Your model will be tested on separate iris images. For testing, you have to code in such a way that it should provide predicted masks for the given input images. Your task is to write script to save predicted mask for each testing image in a predicted_mask folder.
- Zip the code and model file as **1.zip**.

3 Core Point Detection in Fingerprints

3.1 Problem Statement

In this question your task is to find "Core Point" on a fingerprint image captured through different sensors. Core point refers to the centre point of a fingerprint image and is found in the middle of spiral(Read more about Core point on Internet). You will be provided with fingerprint images and corresponding Core point ground truth. You have to design a Neural Network that can find core point coordinates for you given fingerprint image as input.

3.2 Training Phase

Training set consists of 4000 images and their corresponding ground truth.
Input: Your code should take a folder as a input which contains separate folders for both training images and ground truth. Your code should run like this:
$\gg python3corePoint.py --phase = train --epochs = 100$
Enter the training folder: "path_to_training_folder"
and it should start training for 100 epochs (Try to save weights after every epoch.).

3.3 Submission

Your model will be tested on 1000 images that are not given to you. To test your code should take test images folder as input and output the Core point corresponding to each image file in a separate folder(with file name same as the images file).
$\gg python3corePoint.py --phase = test$
Enter the testing images folder: "path_to_testing_folder"
It should output your predictions for each file in a separate folder. We'll run a script to calculate accuracy of your model. Hence keep the file name of predictions file same as image file(apart from extensions). You are first advised to plot the corresponding ground truth on the image to better understand the problem. In case of any clarification regarding question contact TA's.