COMP6721 Applied Artificial Intelligence (Fall 2020)

Project Assignment Part I

Due date : Friday, November 20th

AI Face Mask Detector

Team Name: SS\_G11

Team members:

Lin Li 40044486

Wenhui Guo 学号！！！！

*Index*

*0. Introduction*

*1. Dataset*

*2. CNN Architecture*

*3. Evaluation*

*4. Reference Section*

0. Introduction

In this project, we developed a Deep Learning Convolutional Neural Network (CNN) using PyTorch and train it to recognize three different classes:

(1) Person without a face mask,

(2) Person with a face mask, and

(3) Not a person (i.e., any other image).

In this first phase of the project, we focused on a proper design of our datasets, collecting suitable training data, setting up the complete AI learning & evaluation process and gathering first results. We will further improve it in the second phase of the project.

1. Dataset

Describe how you built your dataset and where you collected images (provide details on each image's source in a file). Provide statistics on the size and structure of your dataset, i.e., how many images you have in each class.

Length: ca. 1 page

Training Data. Create datasets for training and testing your AI. You have to provide provenance information, i.e., where you obtained each image in your dataset. You can re-use existing datasets, but again please make sure you properly reference the source of the images (name, author, source, license of the dataset). Also, note the additional evaluation task that will follow in Part II of the project mentioned below when setting up your dataset.

(I) Data Specialist, responsible for creating, pre-processing, loading & analyzing the datasets; （这两点你可以留着给我，你尽量写你能写的）

Deliverables:

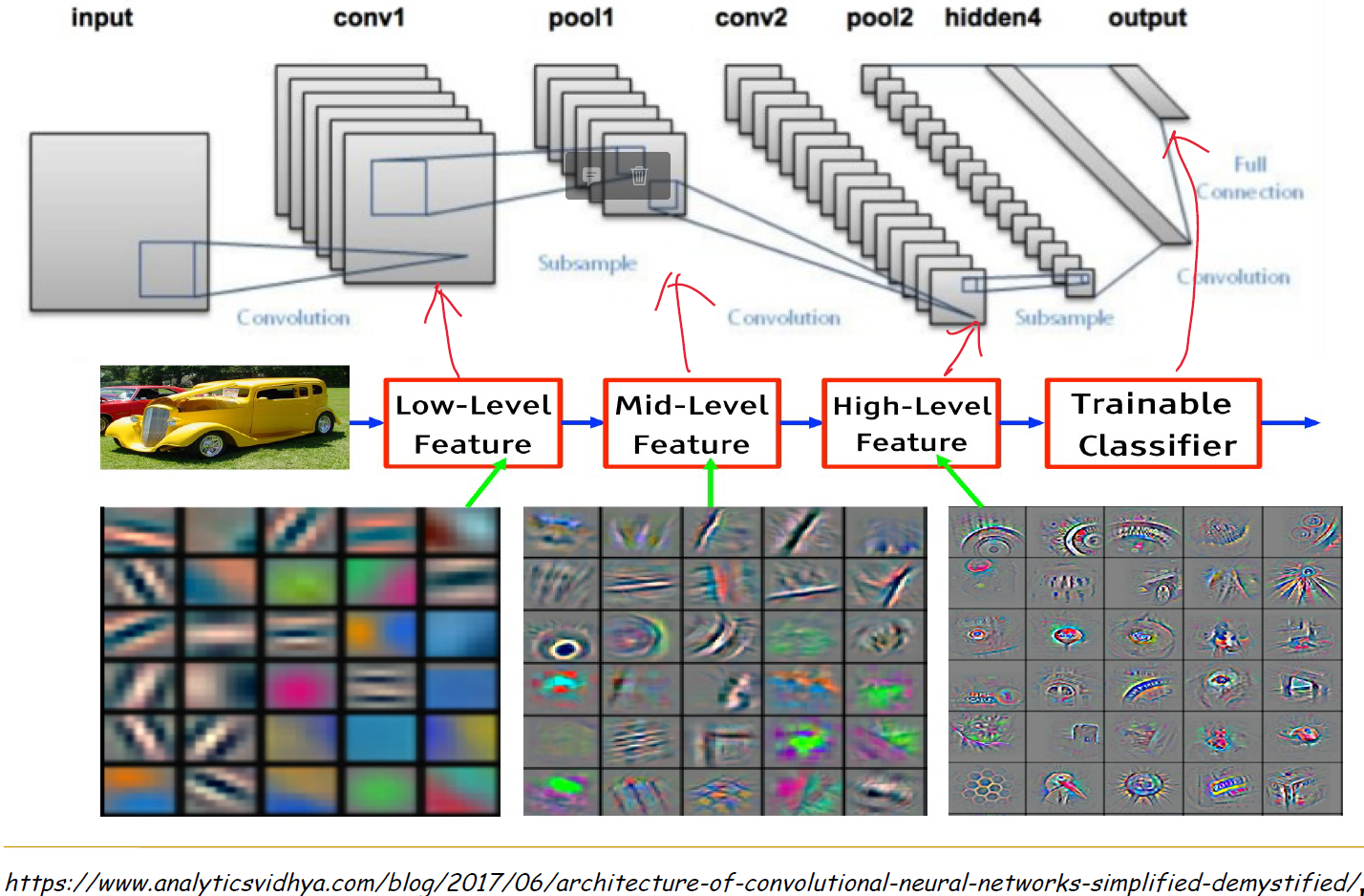
Dataset: The dataset you collected, as well as a fille detailing the source of each image.

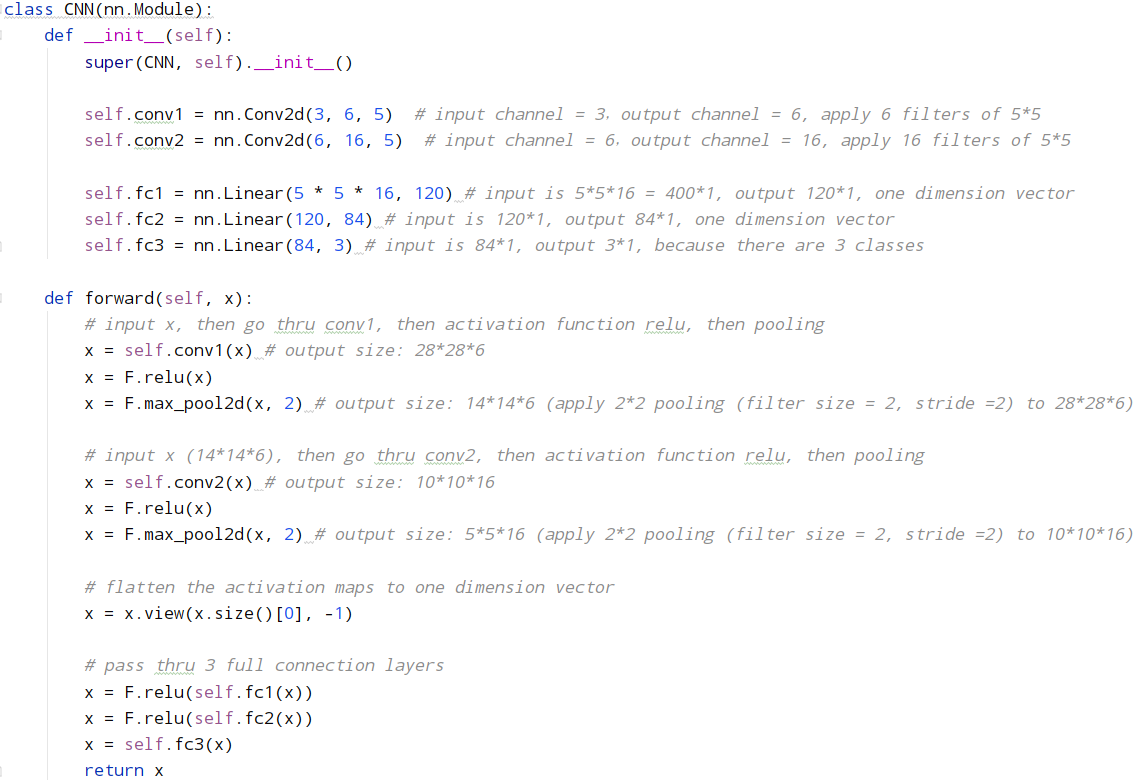
2. CNN Architecture

In this part we will describe the architecture of our CNN and provide details on the training.

In order to create a suitable Convolutional Neural Network (CNN) architecture, implement it in PyTorch, and train it using our dataset, we first create a class inheriting from the nn.Module to define different layers of the network based on provided network architecture above.

We set 2 Convolutional Layers with activation and max pooling, the flatten it to one-dimension vector, then pass it through 3 Full Connection layers, like shown in the teaching material:





In the training phase, we use “train\_loader” to load the images from given path, the images in each sub-folder is labeled with the sub-folder name.

Following what we learned from the lecture and lab, we create an instance of the Convolution class we defined in previous part, then define the optimizer and loss function.

We set the learning rate to be 0.001, this number is small so the weight won’t get changed violently.

We train the model for 10 epochs, for now we can get a final accuracy of ???. (需要全部3000个图片都好了之后再跑出来数字

In second phase of the project we can increase this number to achieve better performance.

When training phase is completed, the parameters of our CNN are saved to 2 files:

net.pkl

net\_params.pkl

This will be read by the testing phase to evaluate our model.

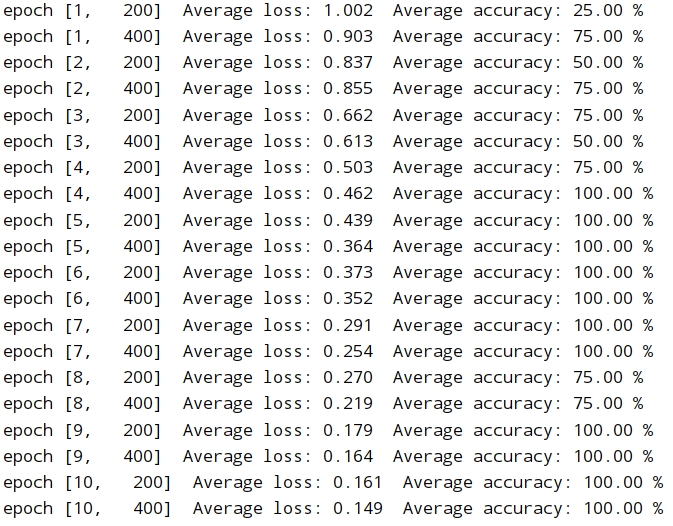
3. Evaluation

We provide the data as below to analyze and evaluate our model:

(1) Log for training our CNN:

Average loss and accuracy of each 200 images

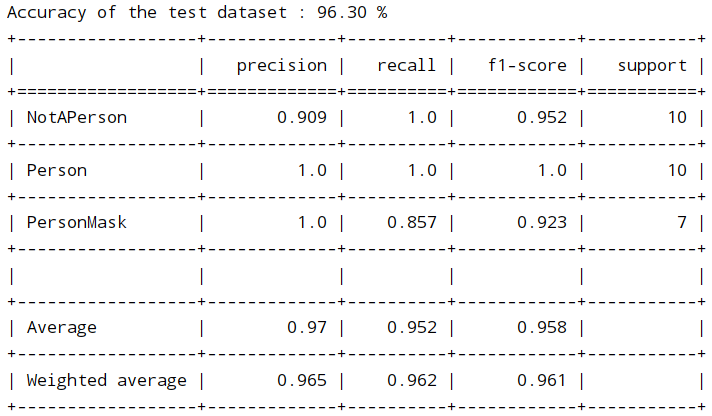
(需要全部3000个图片都好了之后再跑出来数字，到时要重新粘贴截图)



It is obvious that along with the process of training, the average loss decreases, while average accuracy is augmented.

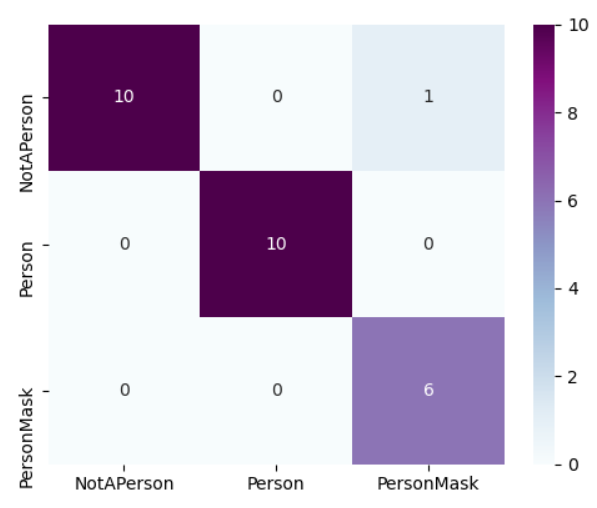
(2) A table of results showing the accuracy, precision, recall and F1-measure

(需要全部3000个图片都好了之后再跑出来数字，到时要重新粘贴截图)



(3) Confusion matrix:

(需要全部3000个图片都好了之后再跑出来数字，到时要重新粘贴截图)



(4) Discuss the results and explain how and where we want to improve during the second phase of the project.

- We could improve on the structure of our CNN:

* For structure:
* We could increase the number of Convolutional Layers so that the accuracy will be further improved;
* We could add normalization to some steps in Convolutional Layers;
* We could add the step of “dropout random nodes” in Full Connection Layers to avoid certain feature has huge impact on the result;
* For parameters:
* We could change the kernel size of the Convolutional Layers (currently always 5\*5);
* We could add padding (currently no padding);
* We could try other activation functions (currently relu);
* For training phase:
* We could try different loss function (currently using CrossEntropyLoss());
* We could use other optimizer (currently using optim.SGD);
* We could change the learning rate to see if the performance can be further improved (current learning rate = 0.001)
* we could increase the number of images in training set, until it reaches the top of the learning curve (where accuracy hardly could be improved by increasing the size of the training set).
* Also We could increase the number of epochs to minimize the loss, and maximize accuracy, precision, recall and F1-measure.

4. Reference Section

This part contains citations to all relevant resources that we have consulted i, even if it was just to inspire us.

Data part:

[]

[]

[]

[]

[]

[]

[] 写完麻烦编一下号码（全部第4部分）~~

[] Documentation of pytorch, <https://pytorch.org/>

[] Build our own datasets and load to pytorch, <https://pytorch.org/docs/stable/torchvision/datasets.html>

[] Lab 7 of COMP6721, refer to moodle

[] https://www.youtube.com/watch?v=YRhxdVk\_sIs

[] Youtube video: How Convolutional Neural Networks work, <https://www.youtube.com/watch?v=FmpDIaiMIeA&ab_channel=BrandonRohrer>

[] “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way”, <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

[] Precision, recall, and F1measure, From Wikipedia: <https://en.wikipedia.org/wiki/Precision_and_recall>

[] A Comprehensive Tutorial to learn Convolutional Neural Networks from Scratch (deeplearning.ai Course #4), <https://www.analyticsvidhya.com/blog/2018/12/guide-convolutional-neural-network-cnn/>

[] Documentation of pytorch: TRAINING A CLASSIFIER, <https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>

[] Neural Network Programming - Deep Learning with PyTorch, <https://deeplizard.com/learn/video/0LhiS6yu2qQ>

[] Documentation of pytorch , <https://pytorch-lightning.readthedocs.io/en/latest/metrics.html>

[] Architecture of Convolutional Neural Networks (CNNs) demystified, <https://www.analyticsvidhya.com/blog/2017/06/architecture-of-convolutional-neural-networks-simplified-demystified/>