COMP6721 Applied Artificial Intelligence (Fall 2020)

Project Assignment Part II

AI Face Mask Detector

Team Name: SS\_G11

Team members:

Lin Li 40044486

Wenhui Guo 40092514

*Index*

*I. Project Updates*

**1. Dataset**

**2. Image normalization**

**3. Visualize CNN performance by showing image with its expected label and predicted label**

**4. CNN Architecture improvement**

*II. K-Fold Cross Validation*

**1. Definition**

**2. Implementation**

**3. Evaluation**

*III. Reference Section*

*I. Project Updates*

1. Dataset

1.1 increase size of dataset

1.2 split dataset into training (80%) and testing (20%). Out of the 80% training dataset, do a 10-fold cross validation

**2. Image normalization**

As pointed out by Mr. Soroosh Shahtalebi in Project Part I demo, we optimized our image process for its normalization.

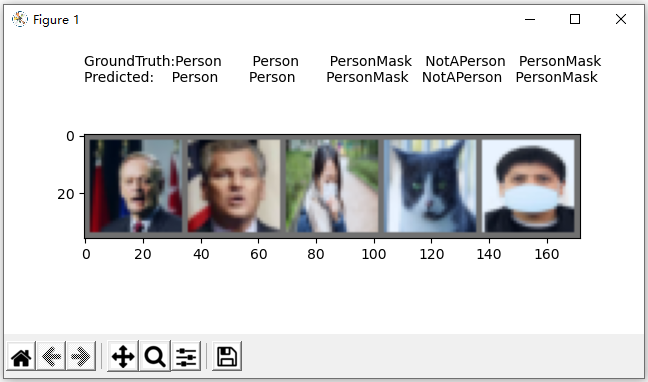
When load images with method torchvision.datasets.ImageFolder, during transform, we pass in mean and standard deviation to method torchvision.transforms.Normalize(mean, std), normalize each channel of the input image (torch.\*Tensor) with formula output[channel] = (input[channel] - mean[channel]) / std[channel]. With this step, we normalize the dimensions of RGB (3 channels) image to [-1, 1].

In the testing phase, to visualize the performance of our CNN, we need to show the images together with their expected class and predicted class, so un-normalization is necessary to transform the image bac to its RGB value. We calculated the corresponding RGB value of images [0, 255] in method def imshow(), then visualize it by plt.imshow() and plt.show().

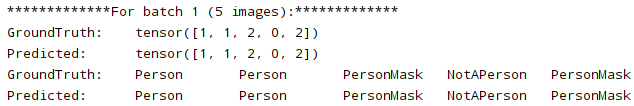
**3. Visualize CNN performance by showing image with its expected label and predicted label**

In Project Part I, during test phase, we managed to print the expected labels and predicted labels, one batch size at a time, in console, for comparison and evaluation.

As suggested by Mr. Soroosh Shahtalebi, we further improved this during the Project Part II, by adding the function of showing the images by methods plt.imshow() and plt.show(), with printing the expected labels and predicted labels beside each image.



In the meantime, this information is printed to console too:



The approach of showing image with its expected label and predicted label facilitate visualization of the CNN performance, it provides an opportunity for us to evaluate the architecture and improve.

**4. CNN Architecture improvement**

improve architecture according to evaluation

***4.1 CNN class for Convolutional Neural Network architecture***

Add hidden layer

Add more convolutional layers

Increased accuracy

Code还没做，看看有没有时间吧

*II: K-Fold Cross Validation*

**1. Definition of K-Fold Cross Validation**

K-Fold Cross Validation is a resampling procedure used to evaluate machine learning models on a limited data sample. It generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split. Importantly, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times.

Process of applying K-Fold Cross Validation:

(1) Shuffle the dataset randomly;

(2) Split the dataset into k consecutive groups, called folds, of equal sizes (if possible).

(3) For each unique group:

Take the group as a hold out or test data set;

Take the K-1 groups as a training data set, the remaining 1 group as testing data set;

Fit a model on the training set and evaluate it on the test set;

Retain the evaluation score and discard the model;

(4) Summarize the skill of the model using the sample of model evaluation scores.

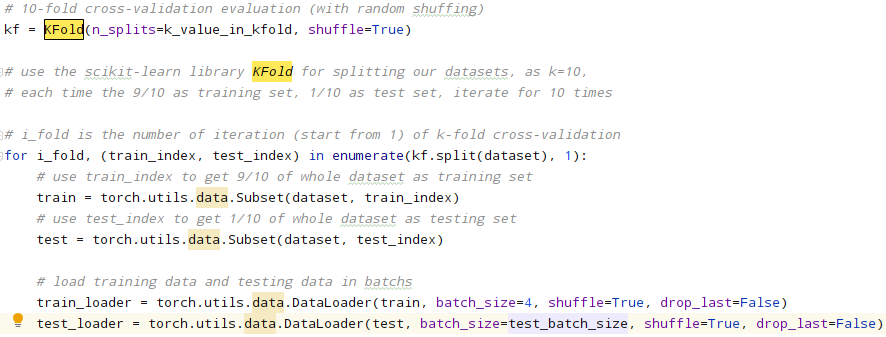
**2. Implementation**

After loading and pre-process images (resize, center crop, normalize, etc.), we use scikit-learn library (sklearn.model\_selection.KFold()) to split our dataset.

With random shuffling enabled, we are able to let our CNN get trained on fixed percentage but random pieces of our dataset, this is very important to lower the system bias.

With K=10 in K-fold, the system uses torch.utils.data.Subset() to assign 9/10 of whole dataset as training data, and 1/10 as testing data.

We use a for loop to enable the 10 iterations of the 10-fold cross validation, where the CNN is trained and tested for 10 times.



For each iteration, we obtain the evaluation measurements including accuracy, precision, recall, F1-measure, and confusion matrix. On completion of the 10 iterations, we calculate the average measurements of all 10 iterations and show the final result.

Dataset好了运行一次，把结果贴过来（因为上面包含图片数量，现在还不能做）

**3. Evaluation**

Compare the results with our previous, standard evaluation (fixed training/test split):

Provide any insights regarding our system's performance in an analysis (i.e., why or why not there is a difference).

During the K-fold, adjusted hyper-parameters, get better result

这个得等TA回复后再改code，再写

*III. Reference Section*

*你再加一些你的reference在最上头，然后给全部编号*

[] Documentation of scikit-learn, <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html>

[] Documentation of scikit-learn, <https://scikit-learn.org/stable/modules/cross_validation.html>

[] Introduction to k-fold Cross-Validation, <https://machinelearningmastery.com/k-fold-cross-validation/>

[] K-Fold Cross Validation, <https://medium.com/datadriveninvestor/k-fold-cross-validation-6b8518070833>

[] Augmenting only the training set in K-folds cross validation, <https://stackoverflow.com/questions/57539567/augmenting-only-the-training-set-in-k-folds-cross-validation>

[] What is the best way to apply k-fold cross validation in CNN?, <https://discuss.pytorch.org/t/what-is-the-best-way-to-apply-k-fold-cross-validation-in-cnn/15035>

[] Documentation of pytorch: TORCHVISION.TRANSFORMS, <https://pytorch.org/docs/stable/torchvision/transforms.html>

[] valid range for imshow with RGB data, <https://stackoverflow.com/questions/49643907/clipping-input-data-to-the-valid-range-for-imshow-with-rgb-data-0-1-for-floa>

[] convert a float32 image to an uint8 image, <https://stackoverflow.com/questions/53235638/how-should-i-convert-a-float32-image-to-an-uint8-image/53236206>

[] Neural Network Programming - Deep Learning with PyTorch , <https://deeplizard.com/learn/video/lu7TCu7HeYc>