Design of experiments

The experiments were carried out by training a CIFAR10 convolution neural network based model with rotated/upright/upsidedows images. The objective was to recognize the orientation of the test images provided. The study was conducted by performing the folowing steps:

1) Preprocessing

The Keras ImageDataGenerator is a tool for both image augmentation and extrating image as numeric vectors straight from a fie sytem. However, there is the requirement of the data to be in a specific folder strucure as shown in the folowing figure.

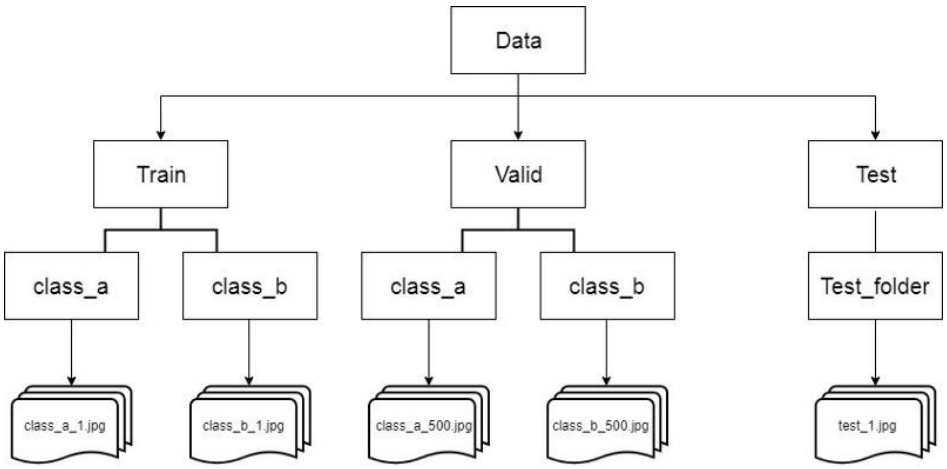


Figure 1: Source: https://medium.com/@vijayabhaskar96

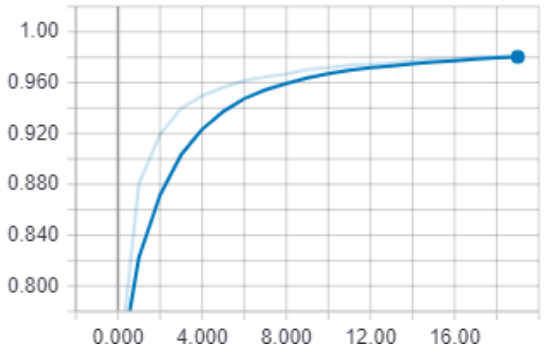
Inside the Train and Validation folders, the data must be splitted in subfolders according to their classes.

Therefore a script called 'PreProcessing.py' was implemented. This script organizes the files extracted from the .zip provided in the git page of the challenge according to the structure described in the Figure 1. The same script also save 10% of the imagens from each class for validation.

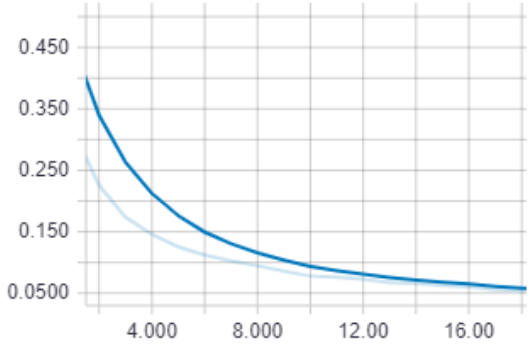
The preprocess file also creates the TensorBoard folder, which will be used to store the information about the training fase os the model and the corrected images fold, which will receive the test images correctly rotated.

2) Training

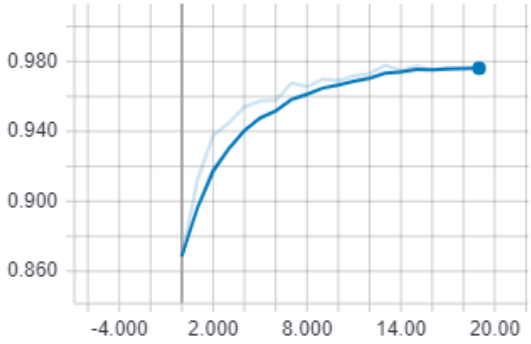
Once the files are organized in the specific fold structure, the 'train.py' script performs the trining of the model by using the Keras ImageDataGenerator tool as both train and validtion inputs. This script also generates a TensorBoardFile inside the 'TensorBoard' fold from which were obtained the following graphs containging the accuracy and loss for both training and validation along the epochs.



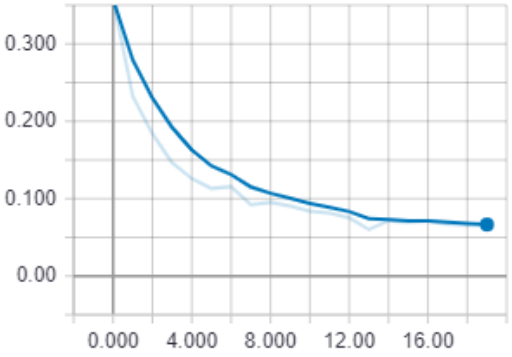
Training Accuracy along epochs



Training Loss along epochs



Validation Accuracy along epochs



Validation loss along epochs

It was observed that after 13 epochs the validation loss/Accuracy stopped evolving, which means that continue training the model after that may lead to a overfitting. So the number of epochs was set to 13 and it was performed another training phase.

After the training phase, the 'train.py' script generates the 'Rot_Faces_Conv_Net.h5' file, which contains the trained model.

3) Testing

The 'test.py' script reads the model from 'Rot_Faces_Conv_Net.h5' file and uses it to classify the images from the test folder. At the end of its execution it writes the 'preds.csv' file containing the name each image from the file along with its respective predicted label. This file along with the 'truth.csv' file can be finally used to measure the performance of the created model over unseen images.

4) Rotate

Finally, the 'rotateImages.py' script reads the 'preds.csv' file in order to rotate the images according to their classification. The objective is to set them all to the upright position. The images are read from the 'test\Test_folder' directory and saved in the CorrectedImages folder in the corrected orientation.