Artificial Neural Networks and Deep Learning 2020

First Competition – Image Classification

Team Members: Leonardo Giusti, Francesco Govigli, Lorenzo Mainetti

SCOPE

Given an image dataset, implement a performant algorithm using CNN (Convolutional Neural Networks) to classify the samples depicting groups of people based on the number of masked people. In particular, the classifier must discriminate between the following images:

1) No person in the image is wearing a mask (class label 0)

2) All the people in the image are wearing a mask (class label 1)

3) Someone in the image is not wearing a mask (class label 2)

Since we have 3 types of images to distinguish we need a 3 class classifier.

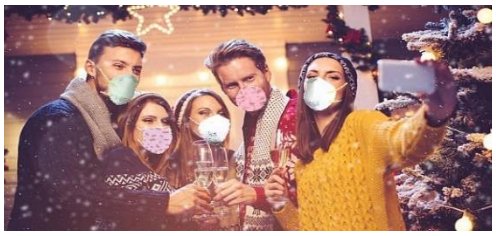
Figure 1. All people are with a mask

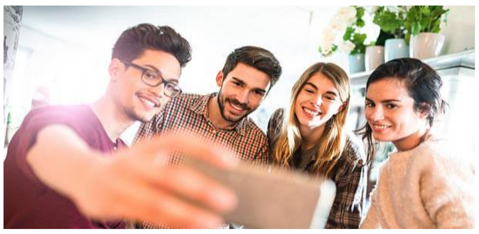
Figure 2. All people are without a mask

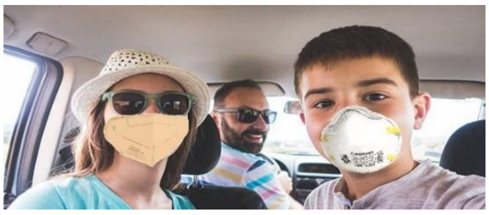
Figure 3. Someone has no mask

MORE DETAILS (Dataset Structure)

The dataset is split in two folders:

- training: 5614 images (class 0 : 1900, class 1 : 1897, class 2 : 1817)
- test: 450 images

Labels for images are provided in the “train_gt.json” file and all the images are of variable size.

FIRST STEP : Dataset preprocessing

Initially, to make all reproducible, we insert a random seed in our notebook.

We used a dataframe to encapsule the data tuple (Image.jpg , label) given in the .json file and splitted it in a training set of 80% of the original set and in a validation set made of 20% of the samples.

Moreover we verify that the data distribution in train_set and valid_set is more or less the same using pandas statistics.

Looking at a first glance to the dataset we see that there are few pictures in it and in addition it looks like pictures are also of different sizes and with some kind of rotations in the features to analyze.

a result, we decided to make a slight Data Augmentation for the training set varying for instance the rotation_range, zoom_range and horizontal flip, but not the vertical flip as it could eventually bring us to an overtrained model.

SECOND STEP : Building a model without Transfer Learing

Model (The baseline):

We started building a model having 4 convolutional layers, composed by a Conv2d, a ReLU activation, a MaxPooling (2x2) and initially 6 filters of size 3x3 with an increasing depth, layer by layer, by two times from bottom to top of the network. The top of the network is build with a Dense layer of 256 neurons and at last 3 neuron layer with a Softmax activation to have an output probability for every neuron representing a class. This model has 3,160,615 parameters and we obtained an accuracy on the test set of 0.55.

Best Model:

We modified our baseline by changing the resize of the input image to 512x512 to reduce loss of quality. Moreover, after looking deeply into the image dataset, we decided to apply more rigid Data Augmentation by increasing: rotation_range = 15, width_shift = 20, height_shift = 20, zoom = 0.2. We tryed also to change the batch_size to 12 and the learning rate to 1e-5 to reach a more precise loss computation and a slowly correction of the trained weights.

The model has just one fully connected layer of 1024 units and a 3 neurons output layer with Softmax activation.

To prevent overfitting we applied Early Stopping with a patience of 10 epochs monitoring the val_loss. The most relevant improvement was given by the 40 filters compared to the 6 initial ones and increasing the convolutional blocks from 6 to 8. This model obtained an accuracy of 0.806 on the test set.

THIRD STEP : Building a model with Transfer Learing

Model (Transfer Baseline):

The baseline for Transfer Learning was made using the VGG-16 Net pre-trained on the ImageNet Dataset. We fine-tuned the network freezing the first 15 layers of VGG-16. We placed at the top a Dense layer made by 512 units with ReLU activation and the 3 neurons output layer with Softmax. We still applied Early Stopping with the parameter restore_best_weights set to True with a patience of 10 epochs monitoring the val_loss. Our baseline afforded 0.83 on the test accuracy.

Best Model:

We still used VGG-16 pre-trained Net, but we froze a smaller number of layers (only 6) to have a more accurate trained network. We also added a GAP layer, two Dense layers (1024 and 256) and a two Dropout layers with a probability of 0.5 at the top of the network. The quantity monitored by the Early Stopping was changed to val_accuracy. We obtained an accuracy on the test set of 0.96.