II.Milestone

The data

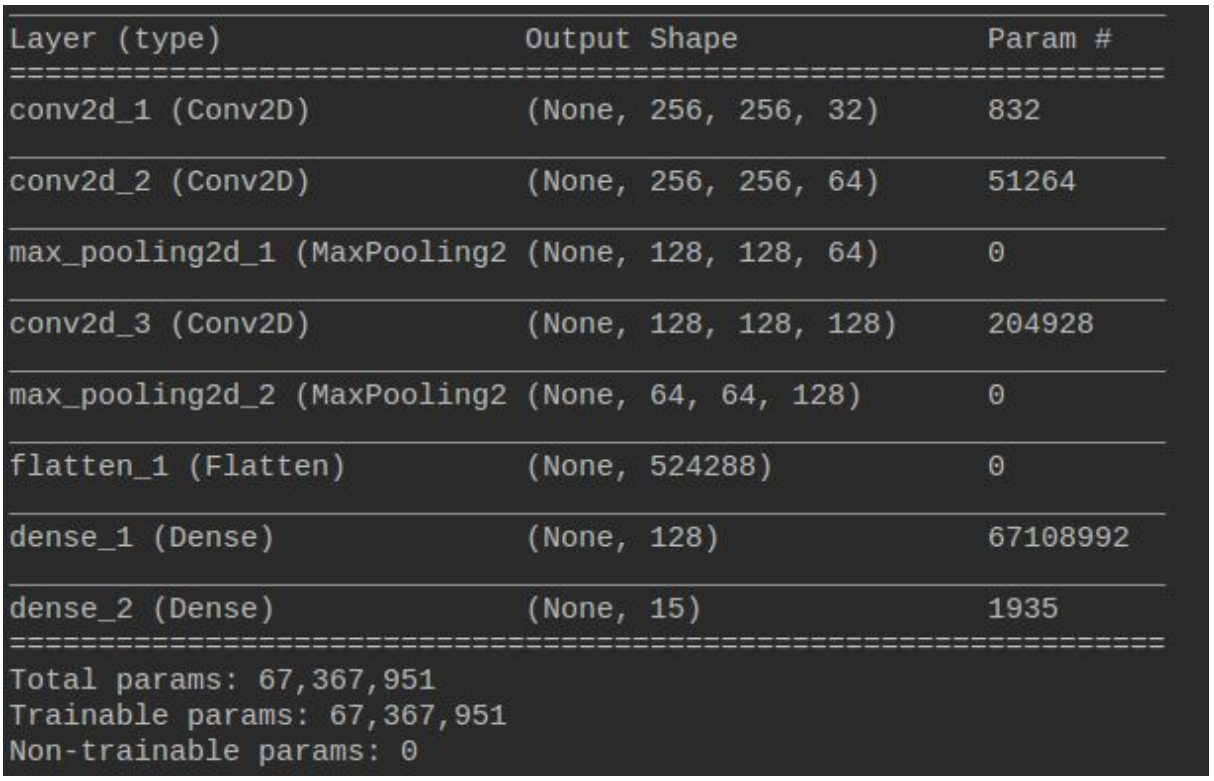
We modified our image preprocessing code, because it didn't fit with the model, and we got error messages. Now it's in the same file with the model (*model.py*). For the first trainings we used only the first 4176 pictures of the dataset, because our dataset is too large to testing. The name of the new dataset is *data_first_folder.csv* .

The model

We used Keras with Tensorflow backend. For our project we created a Convolutional Neural Network (CNN). It has three convolutional, two max-pool and two fully-connected layers:

- Conv1:
  - Input: 1@256x256
  - F: 5x5@32
  - S: 1x1
  - Z: same
- Conv2:
  - Input: 32@256x256
  - F: 5x5@64
  - S: 1x1
  - Z: same
- MaxPool1:
  - 64@256x256
  - F:2x2
  - S:2x2
- Conv3
  - 64@128x128
  - F: 5x5@128
  - S: 1x1
  - Z: same
- MaxPool2:
  - 128@128x128
  - F:2x2
  - S:2x2
- Flatten
- Dense1:
  - 128 neurons
  - activation: tanh
- Dense2:
  - 15 neurons, because we have 15 classes.
  - activation: softmax

We used tanh as an activation function in every layer except last one, there we used softmax.



Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 256, 256, 32)	832
conv2d_2 (Conv2D)	(None, 256, 256, 64)	51264
max_pooling2d_1 (MaxPooling2	(None, 128, 128, 64)	0
conv2d_3 (Conv2D)	(None, 128, 128, 128)	204928
max_pooling2d_2 (MaxPooling2	(None, 64, 64, 128)	0
flatten_1 (Flatten)	(None, 524288)	0
dense_1 (Dense)	(None, 128)	67108992
dense_2 (Dense)	(None, 15)	1935

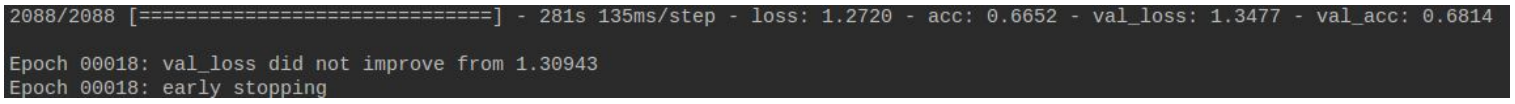
Total params: 67,367,951
Trainable params: 67,367,951
Non-trainable params: 0

1. Summary about the layers and params

Training

We used some Keras callback functions:

- EarlyStopping: The patience is 10 epochs. We defined 100 epochs, but the model stopped at the 18th.



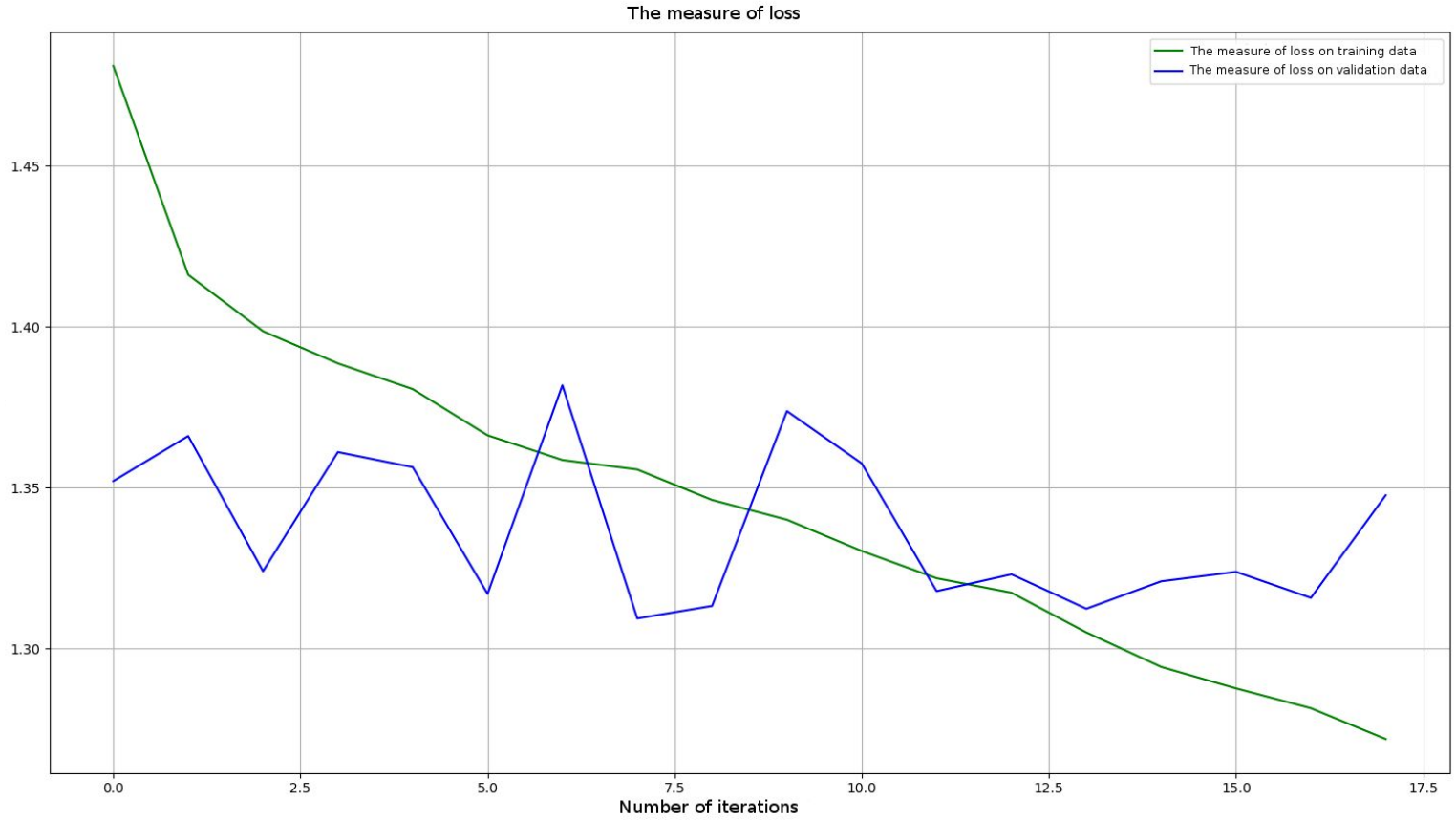
```
2088/2088 [==============================] - 281s 135ms/step - loss: 1.2720 - acc: 0.6652 - val_loss: 1.3477 - val_acc: 0.6814

Epoch 00018: val_loss did not improve from 1.30943
Epoch 00018: early stopping
```

2. Training stopped at 18th epoch

- ModelCheckpoint: It monitors the validation loss and save the best model to *weights.hdf5.*
- ReduceLROnPlateau: It reduces automatically the learning rate.

The data is separated in batches which size is 20.



3. The measure of loss

Testing and evaluation

The best mean-squared error is 0.037786, the best validation loss is 1.30943, and the best accuracy is 0.6652. The model didn't learn well enough. There are more reasons why. First, we used just a little part of the whole dataset (3,7%). It's possible that the model didn't get sample of every classes during the training phase, but during the validation or test phase did. We can improve the training if we add features to the input data, like the patient's age, gender etc.

The next step is to execute our code on AWS with the whole dataset, and then trying out are model with the added features.