Second Homework Machine Learning

Edoardo Di Paolo 1728334

December 2020

Contents

1 Introduction

The task of this homework is to classify different images in 8 classes. In my case I've these classes:

- All purpose cleaners;
- Apricots;
- Crackers;
- Olives jar;
- Soft drink can;
- Cereal bowl;
- Plastic bag;
- Plastic spoon.

2 Data preprocessing

Initially the dataset was composed of 8 different folders corresponding to each class. I had to split the dataset into two folders: one folder is about the training and the other one is about the validation. To do this, I used a python library called *splitfolders* in which I specified the different percentage: 70% for the training and reamining 30% for the validation. After this, I started the real work. To train my neural network, I had to create two *ImageDataGenerator*, one for the training and the other one for the validation. With the ImageDataGenerator I can apply some data augmentation on the dataset (training one in particular). This process consists of modifying images by applying for example some geometric transformations, flipping, color modifications, cropping and rotations. We have 6192 images for the training and 2657 for the validation, for a total of **8849** images in the dataset.

3 Evaluation Models

In this section I'll write about the different neural networks I used to evaluate my dataset. In particular I used two very known CNN and two custom CNN.

3.1 AlexNet

AlexNet is the first model that I used. This is a CCN, a *convolutional neural network*, where there are different layers:

- three convolutional layer with *ReLU* activation function followed by a *MaxPooling* layer and a *BatchNormalization*;
- two fully connected layyers with one of them followed by a *Dropout*.

In the last fully connected layer, the activation function used was the *softmax* and not the *ReLU*.

3.2 VGG16

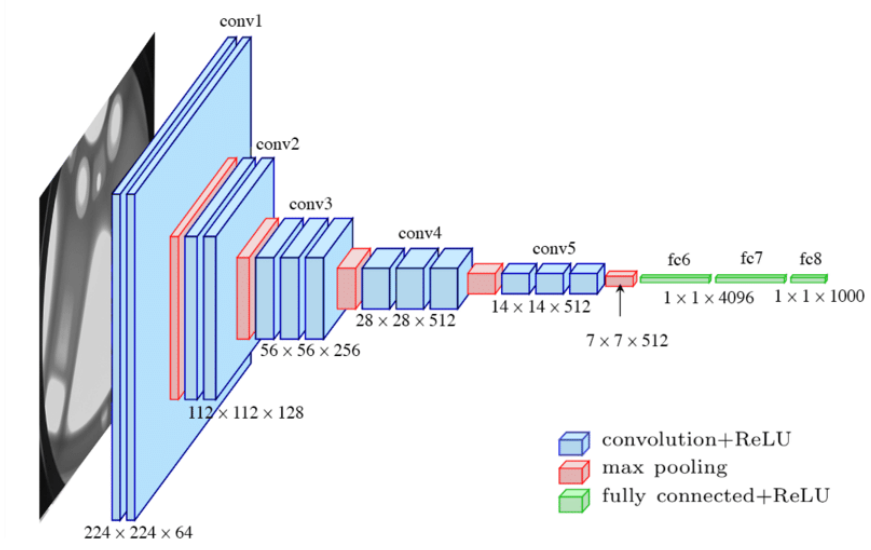
VGG16 is the second model that I used. Also VGG16 is a CNN, but it is really different from AlexNet. In this model we have these layers:

- two convolutional layer (with different filters number) and one max pool layer (this for 4 times);
- two fully connected layers with 4096 units each one;
- one fully connected layer with 8 units, this is the last layer.

In this model also was used the *ReLU* activation function except for the last layer where was used the *softmax* function.

To give an idea about the network's type, I want to show this image that explain better the different layers that it has.

Figure 1: VGG16 Model.



3.3 First custom model

This is the first of two models that I created. This models consists in these layers:

- one convolutional layer and a max pooling layer (for 3 times);
- a *Dropout* on the network;
- two fully connected layers, one with 128 units and the last one with 8.

Also in this case I used *ReLU* activation function except for the last one where I used the *softmax* function.

3.4 Second custom model

This is the second model that I created. In this moddel we have different layers:

- one convolutional layer followed by a maxpooling layer and a *dropout* (this for 3 times);
- three fully connected layers, one with 128 units, one with 4096 and the last oone with 8.

Also in this case I used *ReLU* activation function except for the last one where I used the *softmax* function.

4 Models results

In this section are explained the results of each method that I tried. Each method was tested with 25 epochs. I saw that after 25 epochs the accuracy didn't grow too much, so I decided to stop at 25.

4.1 AlexNet results

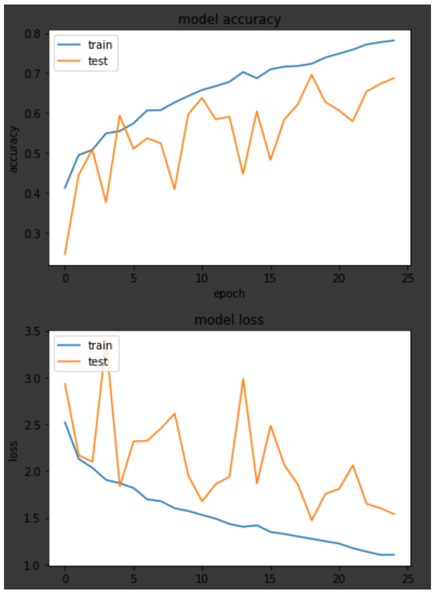
This model was trained (and tested) with a learning rate equal to 0.0001. For this model I got an accuracy equal to 0.68, that is very good but with a loss equal to 1.53.

To evaluate correctly the method, I used the different metrics like the precision, the recall and the F1-score.

Each metric is summarized in the following table with the results that I obtained for each class.

	precision	recall	f1-score	support
All Purpose Cleaners	0.914	0.716	0.803	342
Apricots	0.758	0.725	0.741	345
Crackers	0.461	0.854	0.598	362
Olives jar	0.786	0.777	0.781	354
Soft drink can	0.784	0.559	0.653	331
Cereal bowl	0.815	0.721	0.765	337
Plastic bag	0.573	0.578	0.575	327
Plastic spoon	0.747	0.502	0.600	259
accuracy			0.687	2657
macro avg	0.730	0.678	0.690	2657
weighted avg	0.728	0.687	0.693	2657

These are the graphics that summarized the trend of the loss function and the accuracy (on the training set and validation set).

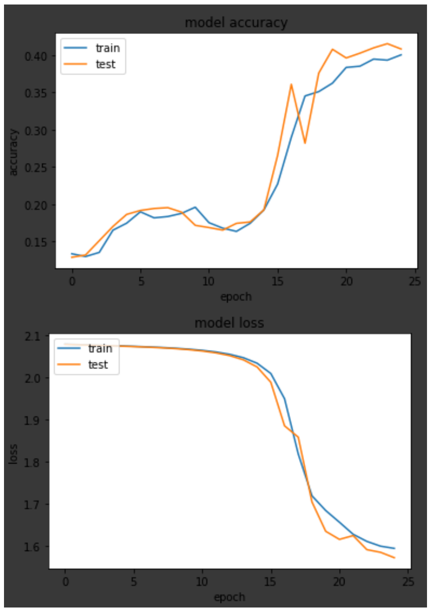


4.2 VGG16 Results

This model was trained (and tested) with a learning rate equal to 0.0001. For this model I got an accuracy equal to 0.408 that is not really good, and a loss about 1.57 that is near the loss of AlexNet. About the metrics, the following table summarize them:

	precision	recall	f1-score	support
All Purpose Cleaners	0.532	0.556	0.544	342
Apricots	0.687	0.330	0.446	345
Crackers	0.314	0.677	0.429	362
Olives jar	0.416	0.576	0.483	354
Soft drink can	0.140	0.045	0.068	331
Cereal bowl	0.507	0.570	0.536	337
Plastic bag	0.334	0.349	0.341	327
Plastic spoon	0.306	0.042	0.075	259
accuracy			0.408	2657
macro avg	0.404	0.393	0.365	2657
weighted avg	0.409	0.408	0.377	2657

We can see how the loss and the accuracy functions are very similar between the training set and the validation set.

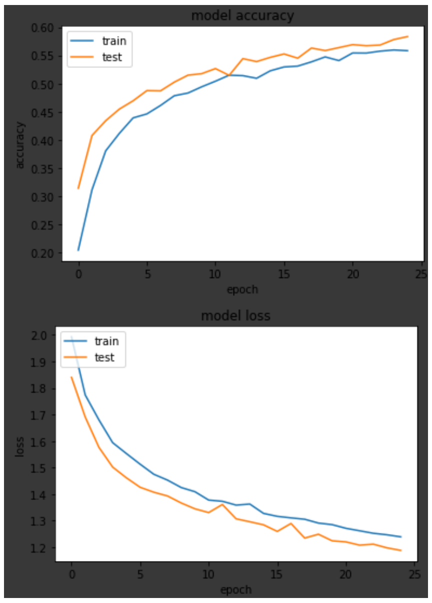


5 First custom model results

In this model I got an accuracy equal to 0.58, with a loss equal to 1.18. This result is better than the one produced by the VGG16 model. The metrics are summarized in the following table:

	precision	recall	f1-score	support
All Purpose Cleaners	0.668	0.681	0.674	342
Apricots	0.706	0.716	0.711	345
Crackers	0.571	0.663	0.614	362
Olives jar	0.714	0.664	0.688	354
Soft drink can	0.453	0.607	0.519	331
Cereal bowl	0.601	0.611	0.606	337
Plastic bag	0.419	0.364	0.390	327
Plastic spoon	0.500	0.266	0.348	259
accuracy			0.583	2657
macro avg	0.579	0.572	0.569	2657
weighted avg	0.583	0.583	0.578	2657

Also in this case the loss and the accuracy between training set and validation set are similar, as shown in the next image.

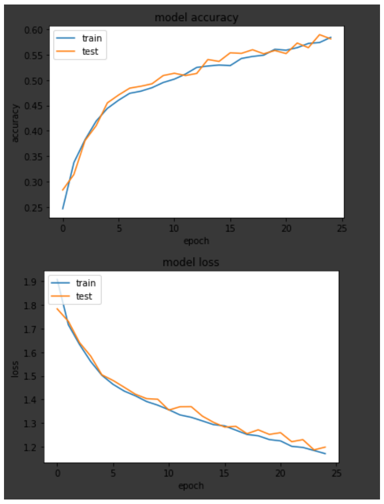


6 Second custom model results

In this model I got an accuracy equal to 0.58, with a loss equal to 1.19. This result is better than the one produced by the VGG16 model and it is similar to the First Custom model. The metrics are summarized in the following table:

	precision	recall	f1-score	support
All Purpose Cleaners	0.666	0.675	0.671	342
Apricots	0.772	0.678	0.722	345
Crackers	0.558	0.555	0.557	362
Olives jar	0.671	0.686	0.679	354
Soft drink can	0.500	0.598	0.545	331
Cereal bowl	0.637	0.608	0.622	337
Plastic bag	0.384	0.526	0.444	327
Plastic spoon	0.504	0.232	0.317	259
accuracy			0.581	2657
macro avg	0.587	0.570	0.570	2657
weighted avg	0.591	0.581	0.579	2657

The next image show the graphic about accuracy and loss function between training set and validation set.



7 Conclusions

In conclusion, I tried different models and different model configurations. In my tests, AlexNet model was the best model that I could find. In the python code, there is also a part about the predictions: it shows to us the error percentage in missclassification between classes. All models, except for the VGG16, have as optimizer adam, while with VGG16 I used SGD due to some accuracy problem with adam. All models required about 30-45 minutes to be trained.