PRACTICUM 3 ARTIFICIAL INTELLIGENCE

Angeli Emilio and Bartoli Ludovico
emilio.angeli@ugent.be
ludovico.bartoli@ugent.be

21 dicembre 2021

1 Question 1

1.1 Question 1.a

The dataset is composed by 20685 training, 10317 validation and 9950 test coloured images of size (100, 100, 3). Each image is stored as numpy.ndarray, with pixels normalized in range [0,1] in float16 (half precision) format. The whole dataset is stored as a (20685, 100, 100, 3) numpy n-dimensional array. Each image is classified in one out of 11 different classes: 'apple', 'banana', 'cherry', 'grape', 'onion', 'peach', 'pear', 'pepper', 'plum', 'potato', 'tomato'. We will assign a number from 0 to 10 to each class according to this order.

1.2 Question 1.b

We observe that in the test, validation and train sets the classes are unbalanced: the classes proportions are not one eleventh each. Anyway, the data set has been splitted evenly:

- Class proportions in the train set:
  $[0.209, 0.046, 0.111, 0.111, 0.044, 0.055, 0.117, 0.057, 0.059, 0.058, 0.132]$,
- Class proportions in the validation set:
  $[0.21, 0.046, 0.111, 0.111, 0.044, 0.056, 0.117, 0.057, 0.058, 0.058, 0.13]$,
- Class proportions in the test set:
  $[0.207, 0.048, 0.115, 0.113, 0.043, 0.058, 0.122, 0.059, 0.055, 0.06, 0.12]$.

1.3 Question 1.c

The following plot shows one sample image per class:

We should scale the image to the correct size and the pixels to the correct values in [0,1] of type float16. Moreover the images must be of type np.ndarray. The background should be white.

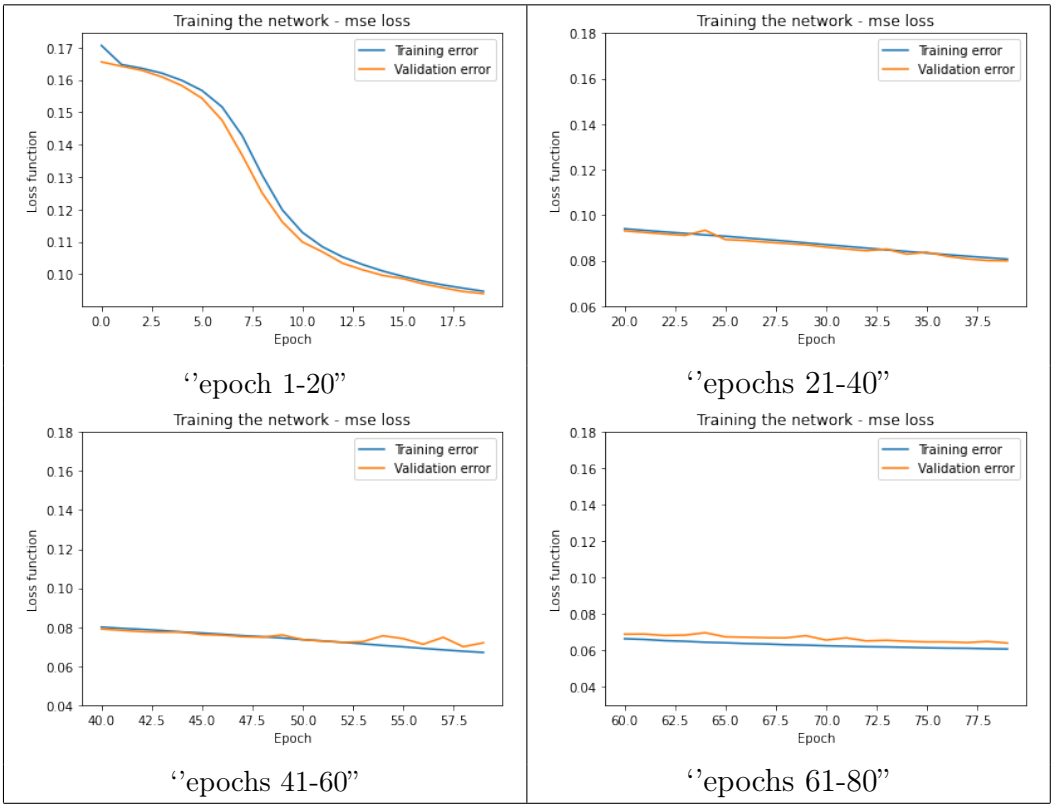
1.4 Question 1.d

Since the fruits occupy the whole space in the image, we can ignore translations. Anyway, we would prefer our classifier to be invariant for rotation and flipping.

2 Question 2

2.1 Question 2.b

In the following plots are shown the learning error curves, for the train and validation data, versus the epochs:



''epoch 1-20"

''epochs 21-40"

''epochs 41-60"

''epochs 61-80"

2.2 Question 2.c

Since the learning curves grow in parallel, we can say the model with the trained parameters fits the data very well (best fit).

3 Question 3

3.1 Question 3.a

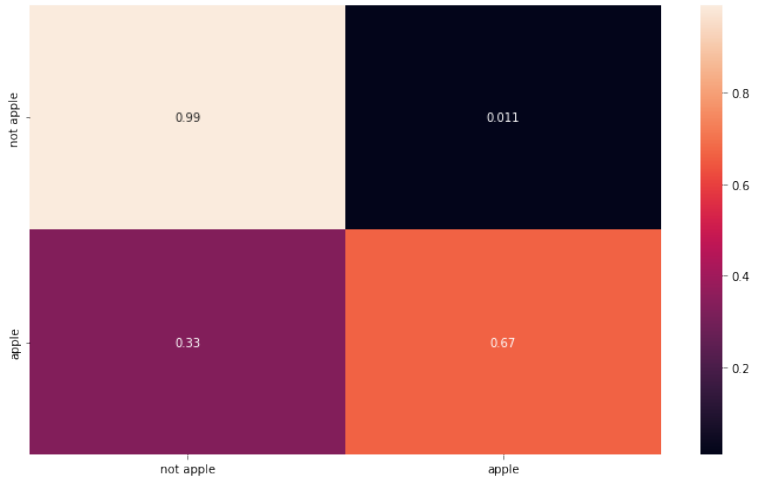
The output represents the probability that the fruit in the image is an apple.

3.2 Question 3.b

After computing the probability of an image to be an apple we have to choose a classification rule. We chose to classify the image as apple if the output is greater or equal than 0.5.

3.3 Question 3.c

We plotted the confusion matrix obtained by applying the model to the test set. Its i-th row and j-th column entry indicates the proportion of samples with true label being i-th class and predicted label being j-th class.



3.4 Question 3.d

In statistical analysis of binary classification,, the F1 score is the harmonic mean (the reciprocal of the arithmetic mean of the reciprocals) of the precision and recall, where the precision is the number of true positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive.

F1 score is used when the False Negatives and False Positives are crucial. Our apple classifier has an accuracy o 0.92% and a F1 score of 0.78%. This is because, ignoring the true negative, the 0.33% of the test apple images that are misclassified as not apples have more weight. We choose F1 score, which ignores true negative, because an apple classifier must be able to classify apples and not other fruits as not apples.

4 Question 4

4.1 Question 4.a

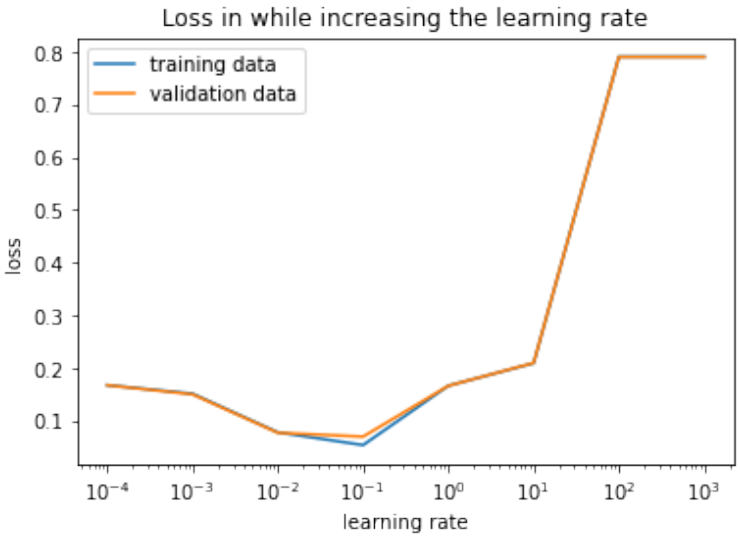
We used Stochastic Gradient Descend as the optimization algorithm to update the weights of our network iteratively. In SGD the weights are updated by the gradient descend algorithm:

$$w_{n+1} = w_n - \gamma \nabla L(w_n)$$

where γ is the learning rate: it is the step used each iteration to update the weights in the direction pointing the (local) minimum of the loss function.

4.2 Question 4.b

Training the model for 20 epochs for different values of γ we obtained the following relationship between the loss error at the end of the last epoch and the learning rate (batch size is always set to 32 images):



As the plot shows, we would use the value $\gamma = 0.1$ because gives the minimum loss using the same number of epochs.

4.3 Question 4.c

When the learning rate is lowest, 10^{-4}, the loss is not too high, just the double of the loss obtained using the optimal lr. The problem of such learning rate is that the loss does not decrease fast enough: since the step size towards the minimum is too little, too many steps will be needed to reach it (with respect to the optimal lr).

4.4 Question 4.d

As γ becomes higher than the optimal one, the loss after 20 epochs grows. Since the step is too high, when the distance between the current parameter vector w_n and the minimum counterimage becomes less than γ, the algorithm is no more able to reach the optimal value of w and the loss does not decrease.

5 Question 5

5.1 Question 5.a

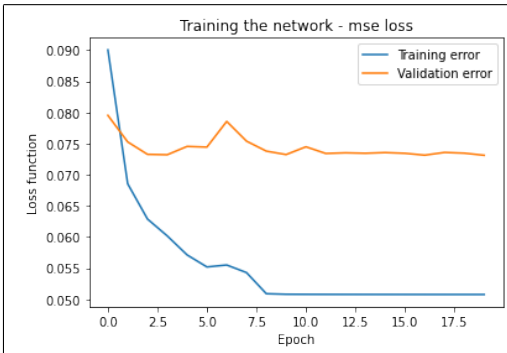
The batch size is the number of samples that will be used in the backpropagation process. Suppose the number of samples on which we want to train our neural network is 1000 and we choose a batch size equal to 50. First, the algorithm will perform a backpropagation procedure considering only the first 50 samples in the train set. Then it will perform another backpropagation procedure using only the samples from the 51-th to the 100-th, and so on. In this way at each iteration the weights are adjourned only using a subset of the train set, the cardinality of this subset (in this case 50) is called 'batch size'. Using this method (with a batch size $<$ size of train set) guarantees a memory and computational complexity reduction, on the other hand it could lead also to accuracy problems: the minimum of the loss function could become harder to reach.

5.2 Question 5.b

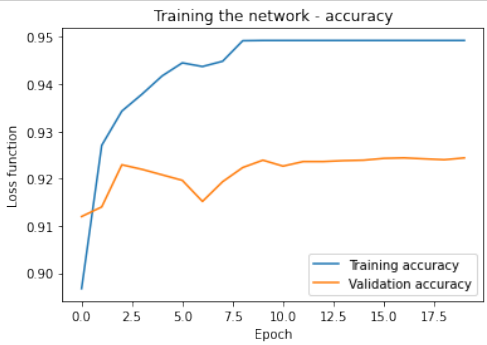
We propose to use 64 as batch size, since with such batch size we obtain the best compromise: an adequate accuracy and a quite low computational time.

5.3 Question 5.c

We fixed the lr to 0.01 and trained the model for different batch sizes. When the batch size is equal to 1, the running time for each epoch increases: we should expect the contrary, this happens because SGD is efficient if the batch size is greater than 1.



''loss curve"

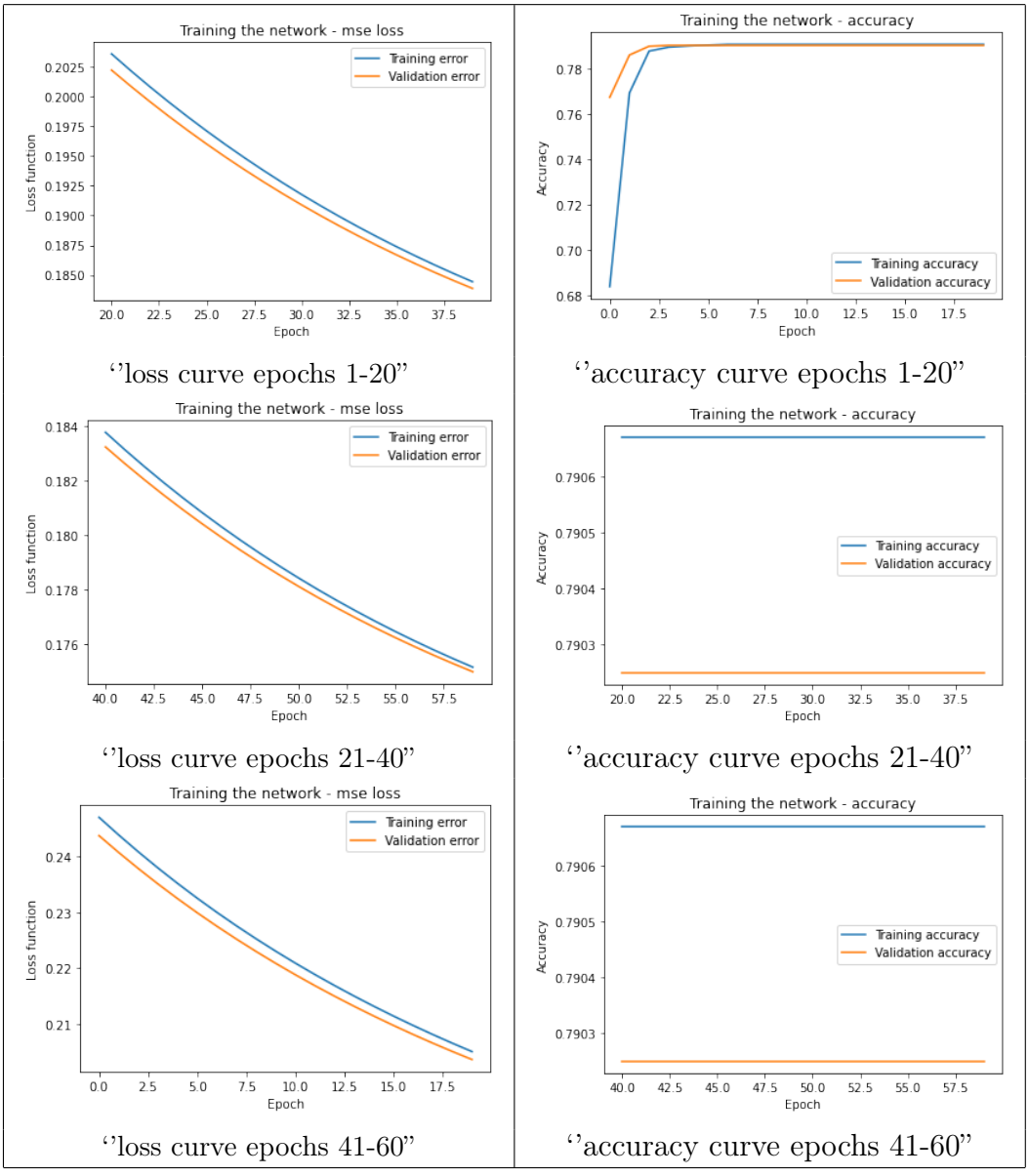


''accuracy curve"

On the other hand, the convergence to a local minimum of the loss function is much faster: after two epochs the loss and the accuracy start to converge.

5.4 Question 5.d

If we set batch size as the entire dataset we get the following results:



''loss curve epochs 1-20"

''accuracy curve epochs 1-20"

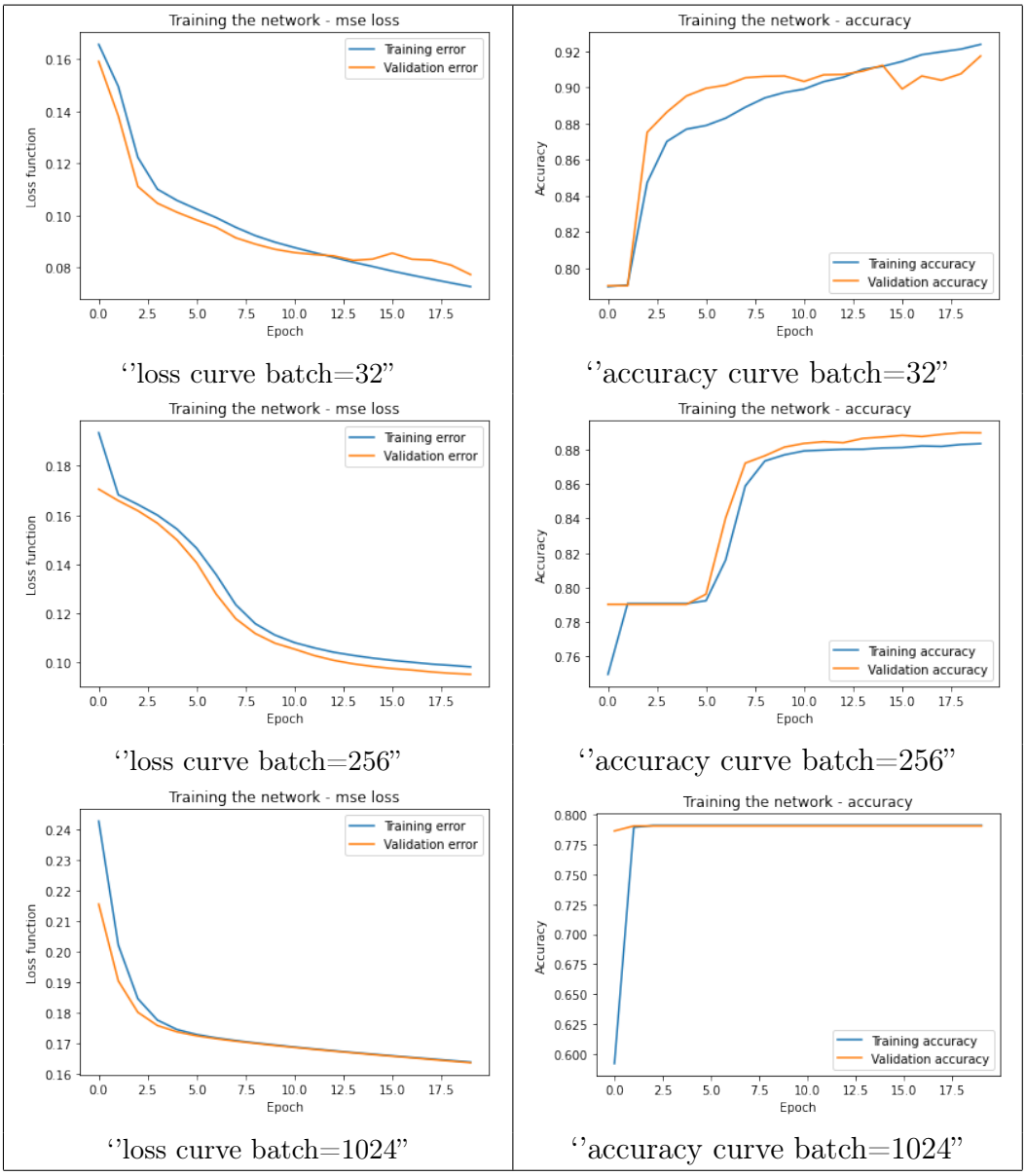
''loss curve epochs 21-40"

''accuracy curve epochs 21-40"

''loss curve epochs 41-60"

''accuracy curve epochs 41-60"

Loss does not converge rapidly but decreases slowly every epoch. The accuracy will decrease only after many epochs. The results will be better in the end but at the cost of a huge training time (a huge number of epochs). We trained the model for 20 epochs for batch size equal to 32, 256 and 1024. The results are the following:



''loss curve batch=32"

''accuracy curve batch=32"

''loss curve batch=256"

''accuracy curve batch=256"

''loss curve batch=1024"

''accuracy curve batch=1024"

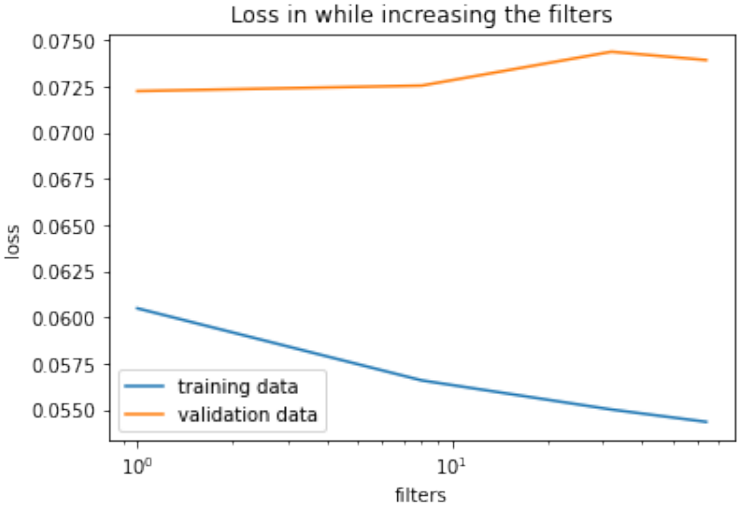
We can see the same behavior of the extreme cases but less pronounced.

6 Question 6

6.1 Question 6.a

We fixed the epochs to 20, the learning rate to 0.1, the batch size to 32 and we trained different models varying the number k=[1,8,32,64] of the different filters of dimensions 4x4 applied in the last convolutional layer.

We evaluated the performance of the models by comparing the validation at the end of the last epoch.



6.2 Question 6.b

We decide to take k = 8 as the best trade off between a low validation and training loss (see question c). The summary of the model (1,301 parameters in total) is as follows:

Layer (type)	Output Shape	Param
average_pooling2d	(None, 50, 50, 3)	0
conv2d	(None, 48, 48, 4)	112
average_pooling2d	(None, 48, 48, 4)	0
conv2d	(None, 22, 22, 4)	148
average_pooling2d	(None, 11, 11, 4)	0
conv2d	(None, 8, 8, 8)	520
average_pooling2d	(None, 4, 4, 8)	0
flatten	(None, 128)	0
Dense	(None, 4)	516
Dense	(None, 1)	5

In a conv2d layer with a kernel of size kxk, the number of parameters is equal to $out\_channels * (in\_channels * k * k + 1)$ where 1 stands for the bias. Note that it does not depend on the size of the images.
In a dense fully connected layer the number of parameters is equal to $(n\_inputs + 1) * n\_outputs$, where 1 bias is counted for each output neuron.

6.3 Question 6.c

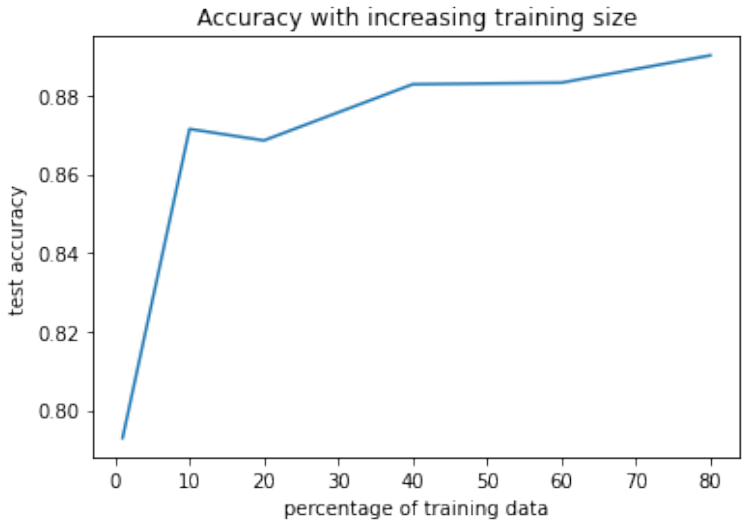
As we can see from the plot, the training loss decreases as the number of features detected in the last layer increases but the validation loss grows. This is because the model is overfitting: in the last layer are detected too many features, which are not class representative but are peculiar of the training images.

If we take k=1 the validation loss is as low as for k=8 but the training loss is too high: the model is underfitted.

7 Question 7

7.1 Question 7.a,b

After random subsampling of training data with different sizes, for each size we trained the model until convergence with lr=0.01 and batch size=64. The resulting test accuracies are shown in the picture:

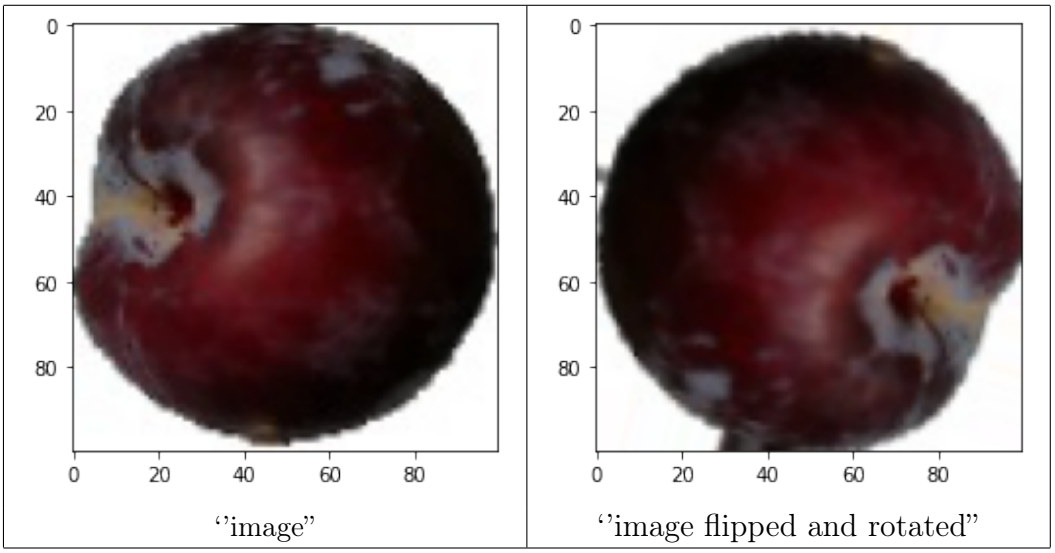


As expected, as the train data are more and more, the network receives more information and its classification performance increases.

7.2 Question 7.c

After selecting a random subset containing 1 % of the data, in order to face a limited amount of training data, we used data augmentation to generate more samples. We applied to each image 9 times both a random flipping (both orizontal or vertical) and a random rotation of 0.08*2*pi degrees. We selected

on purpose a small angle since the white background is no more mantained when bigger angles are used.



''image" ''image flipped and rotated"

We obtained an augmented dataset of length equal to 10% of the whole data. Data augmentation did not give promising results, as the accuracy of the trained model on the test set is still 0.79, as it was using only 1% of the data. Therefore we suggest other solutions, such as using the convolutional part of a network pretrained on ImageNet as feature extractor.

8 Question 8

8.1 Question 8.b

The softmax function is an extension of the sigmoid function to the multiclass case: it transforms a vector in R^K (where K is the number of classes, in our case K=11) into another K-dimensional vector of values between 0 and 1 whose sum is 1. Using the softmax function we will obtain a model that, applied to a new sample, will return a vector whose i-th element represent the probability that the new sample belongs to the i-th class.
The sigmoid function is not applicable here since it can not be used to classify more than 2 classes.

8.2 Question 8.c

Given two (discrete) probability distributions p and q defined on the same support, their cross entropy is defined as

$$H(p,q) = -\sum_{x} p(x) log(q(x))$$

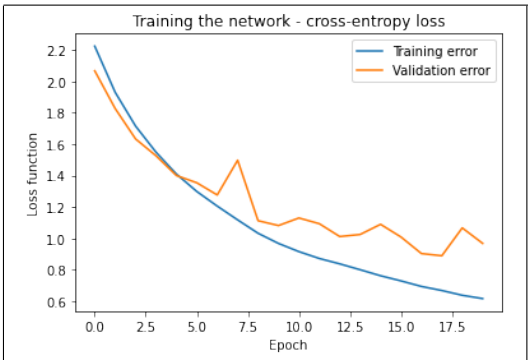
and basically measures how a modelled probability distribution q differs from the true distribution p. In our case, the categorical cross entropy loss is the cross entropy between the predicted probability vector q and the Dirac distribution p, which is a vector of all zeros except 1 for the true class of the image.

Usually, while mean square error is used in regression problems, categorical cross entropy is used in classification problems. This is beacuse the cross entropy 'punishes' more misclassification: if an image is predicted with a low probability to be in its true class, the loss goes to infinity as this probability goes to zero. On the contrary, the mean square error, which is simply the square distance between 1 and the prtedicted probability, remains less than one.

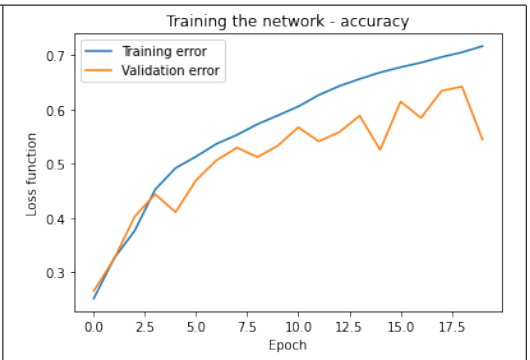
9 Question 9

9.1 Question 9.a

We trained the fruit model for 20 epochs, setting lr=0.01. Using more epochs the model would have overtrained. The learning curves (loss and accuracy) are the following:



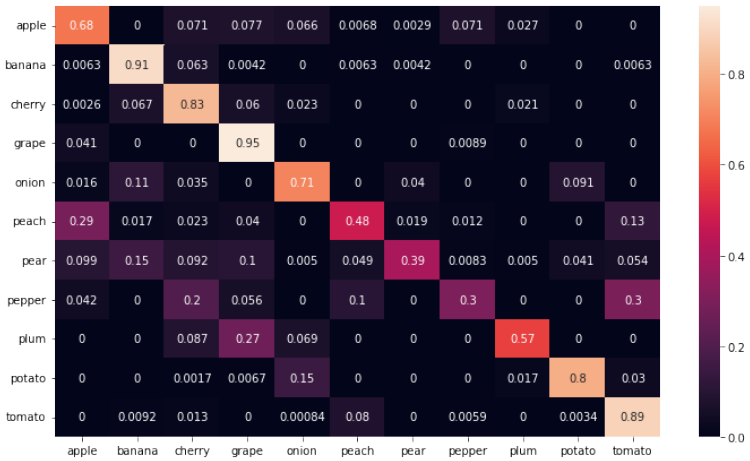
''loss"



''accuracy"

9.2 Question 9.b

The trained model has an accuracy of about 70% on the test data. The following confusion matrix indicates for each class (row) how many images were classified in all the other classes in percentage. Therefore the sum of each row is 1.



We can see that the best accuracy is obtained on the grape, for which the 95 percent of samples are correctly classified.

10 Question 10

10.1 Question 10.a

We collected 31 fruit images from the web: 8 apples, 6 bananas, 6 peppers, 5 potatos and 6 tomatos. Here are some examples:

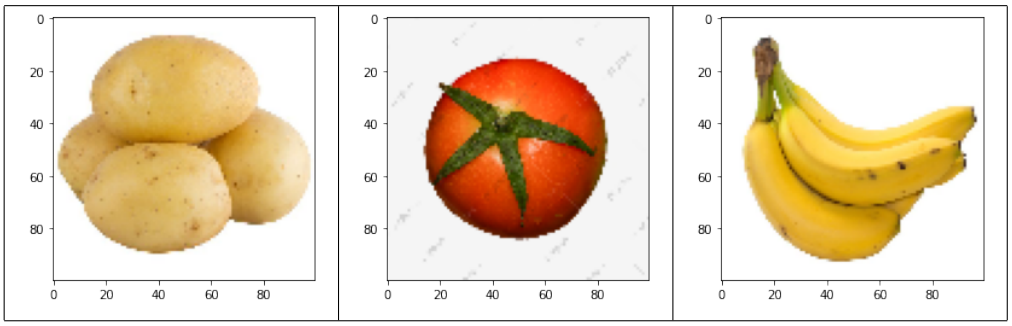
For each of the five classes we downloaded one image containing more than one fruit: one example is the potato image in the picture.

11 Question 11

11.1 Question 11.a

The apple classifier did not perform well at all: none of the observation is classified as apple, as the predicted probabilities are all less than 4%.

The fruit classifier has an accuracy of about 37% on the new data, that is not really high but is much more than that of a random predictor (that would have an accuracy of $\frac{1}{11}$).
Here are some samples which has been predicted correctly:



11.2 Question 11.b

No. Since the fruit classifier classifies almost all the apples as bananas we could think that this is due to the fact that all new images have a larger white background (than the ones of the original data set) and the bananas of the original data set have a large background too. We would suggest to make the images more similar to the training data by cropping them in the center before predicting their class with the model.
Later we added one apple image with less white background and it was classified as a peach by the algorithm, this suggests that probably also other techniques should be tried.
Finally we observe that also the light condition of the image can have an influence: when we try to predict a new image's class we should verify that its light condition matches with the ones of the original dataset.

11.3 Question 11.c

Suppose we would try to classify a new fruit whose label does not belong to any of the 11 classes (for example kiwi). If this neural network would work optimally, then it would give as output the label of the fruit that most resemble the kiwi between the original 11 classes (maybe potato).