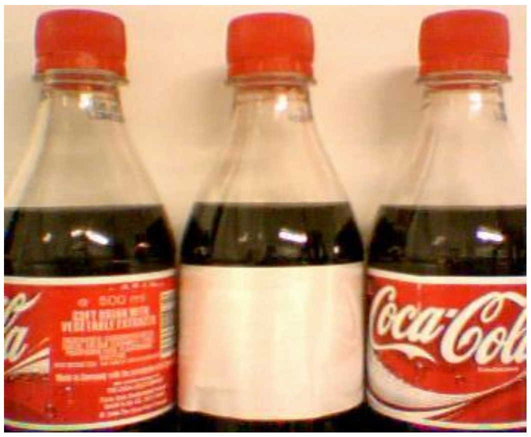
Kuryez Maksim 02.03.98

Production Line Visual Inspection: Soft Drink Bottling Plant



The aim of the project:

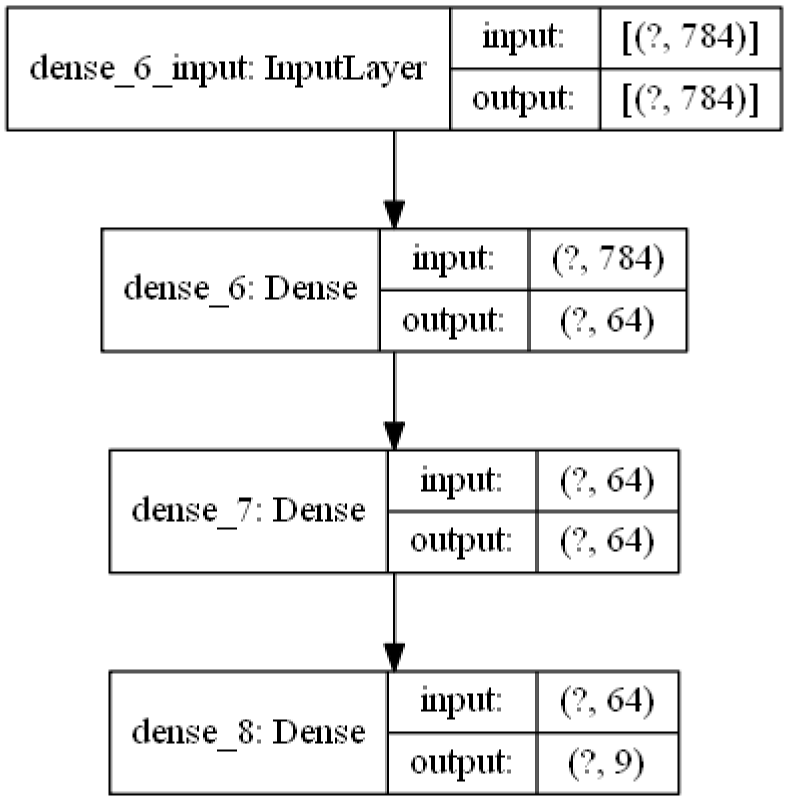
The task is to design and prototype an image processing system to detect the set of fault conditions that may occur together with identifying the type of fault that has occurred.

Here we are dealing with a bottling production line in a facility bottling Coca-Cola. We have a set of images of bottles, leaving the bottling line. The goal is to identify whether the middle bottle is normal or defective, and we need to classify the defect type.

Project content:

1. '/new_bottles': The directory contains the training material, divided in 8 classes: 7 types of defect and a normal bottles class. And the testing material with the same 8 classes.
2. 'Sorting_bottles.py': The tensorflow based deep-learning algorithm which trains the network to determine the defective bottles and classes of defect.
3. 'model.h5': The file contains weights that have been saved after the training. File is produced by the algorithm

4. ‘Data.xlsx’: The comparison of actual data classification and the predictions made by algorithm.
5. ‘model.png’: The graphical representation of network layers structure:



As we can see, for the input we have 1-dimentional array of 784 elements and as the output we get a 1-dimentional array of 9 elements (the same as the number of classes).

6. ‘Accuracy.png’: The graph of network accuracy.
7. ‘Loss.png’: The graph of network loss.

Algorithm tuning:

First, we load the training and testing data from ‘/new_bottles/test’ and ‘/new_bottles/train’ directories. Then using the iterators we create the training and testing numpy image arrays and label arrays.

We reshape the array in order to get 1-dimentional of 784 size (because the size of the picture is 28x28) array for training the data.

Although it is not exactly the 1-dimentional array because we have multiple number of pictures.

Then we initialize and configure the training model the way as we can see in the ‘model.png’ picture.

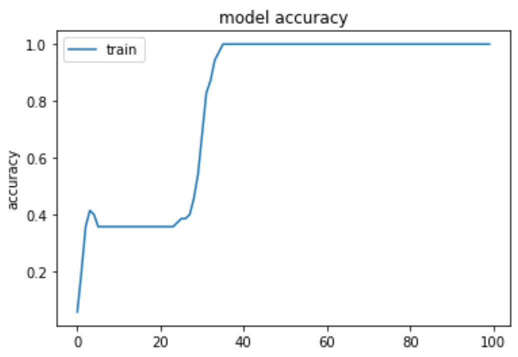
After that we train the model using training data and labels, and evaluated the obtained model (with the obtained weights) using the testing data and labels.

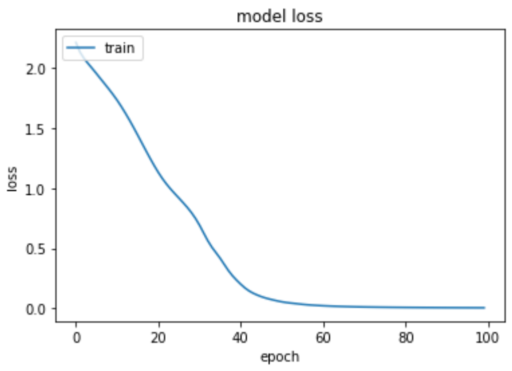
We print the actual classification of testing data (done manually by me) and the predicted classification of testing data (done by algorithm). Usually the accuracy lies within (0.9 - 1) interval.

After that we print the accuracy and the loss plots.

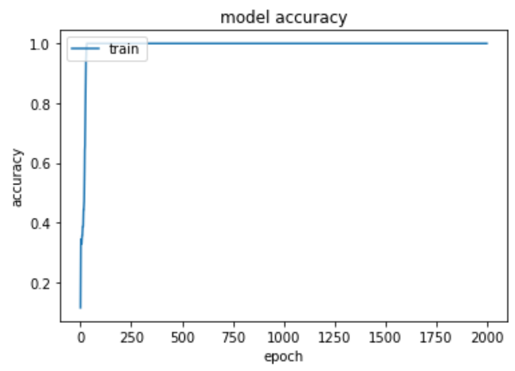
Depending on number of epochs it looks differently.

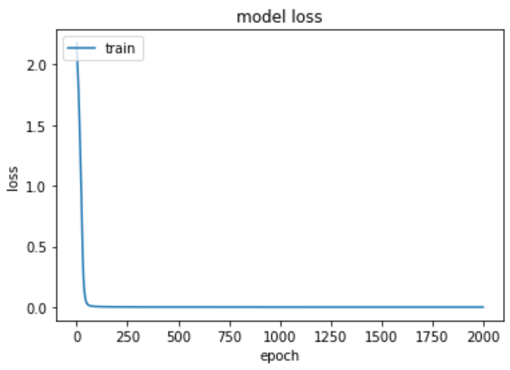
100 epochs:





2000 epochs:





As we can see, it is enough to run approximately 150 epochs in order to achieve reasonable weights for the network. And after that it reaches the top and stops growing.

The conclusion:

The product line visual inspection can be easily achieved by using multilayered deep-learning network. Also it is quite efficient even without a huge number of iterations (epochs), although if you want to use it on production scale you will need to train it with much bigger number of training data, because the lack of training data cannot be compensated by huge number of epochs. Even after 2000 epochs the accuracy is still around 9.5 (You can see it in 'Predictions_Comparison.txt' file).