Università degli Studi di Genova

Robotics Engineering

Sara Romano

Report of Machine Learning course no.5(a+b)

Deep Learning

December 2020

Contents

Chapter 1

Introduction

The fifth assignment deals with Deep Learning. It is splitted into two parts: Autoencoders (5a) and practice with deep learning with some MATLAB tutorials (5b).

The goal of the first part is to train a multilayer perperceptron as an autoencoder; instead the goal of the second part is to acquire practice with the programming style of Matlab's deep neural networks: therefore executing some MATLAB tutorials.

1.1 Autoencoder

An autoencoder is a specific type of feedforward neural network where the input is the same as the the output. It compresses the input into a lower-dimensional code and then reconstruct the output from this. Moreover the code is called "compression" of the input.

The simplest autoencoder has one input layer, one hidden layer and one output layer.

An important property of the autoencoder is that it is an unsupervised learning technique, since they do not need a target to train on. To be more precise it is sometimes called "self-supervised" since the it generates the target from the training data. (the input is the output as well).

Let now discuss a bit on the architecture of the autoencoder: it is composed by three parts: encoder, code and decoder. The encoder and the decoder are feedforward neural networks (ANN). The input goes through the encoder (which has a similar structure of an ANN) and produces the code, which is, at the same way, the input of the decoder. Notice that the same patter as both the input and the target can be used.

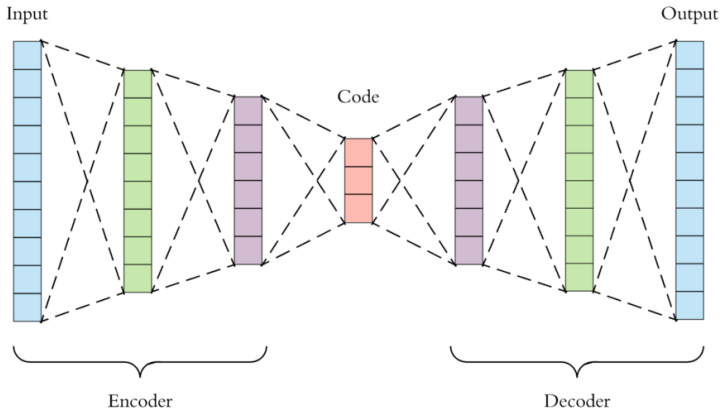

Figure 1.1: Autoencoder Structure

1.2 Deep Learning

Deep learning is a class of machine learning algorithms that uses multiple layers to extract features from raw input. DNNs creates a map of virtual neurons ans assigns a weight for the connections among them. Most modern DNN is the Convolutional Deep Neural Network (CNNs), used in computer vision. The fundamental concept of CNN is, it utilizes convolution of images and filters to produce invariant features that are passed on to the next layer. In the next layer, the features are convoluted with a different set of filters to produce abstract and more invariant features and this process continues till we get final output/feature that is invariant to occlusions.

CNNs has a fixed structure, and the weight on the hidden layer are shared, which means replicated across the layer. It could have different size, depending on the number of layers. The most important layers are:

1) Convolutional Layers: a set of learnable filters (kernels), which have a receptive field (limited-size): each unit has the same weights as the others (convolution kernel), learned from the data.

Here the ReLU (Rectified linear unit) activation function can be seen as a layer: it removes negative values that are not useful for the learning. Usually the max activation function is use, but are also common hyperbolic tangent or the sigmoid fuction.

2) Pooling Layers: placed after convolutional layer, they perform downsampling, reducing the spatial size of the representation, in order to reduce the number of parameters (avoiding over-fitting as well). they work replacing the full information with one scalar.

3) Output Layers: a fully-connected layers: they take the higher-order features and produce class probabilities. The output are typically two-dimensional [b x N] in which b is the num of examples, N is the number of classes.

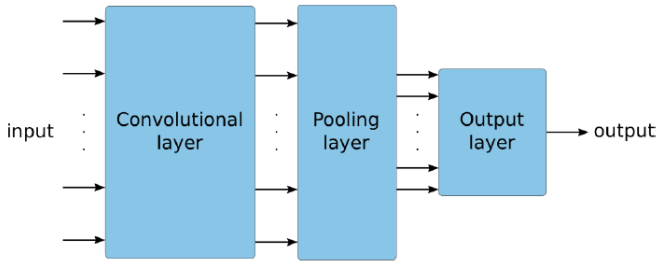

Figure 1.2: CNNs Structure

Summarizing, training a depp network requires a lot of computations, in which we need to use training sets with milions of observations.

Some popular deep networks are available in software libraries, which are pre-trained for specific tasks so any user can employ them without having to do the optimization. Since in the lab we are dealing with image classification, pre-trained models Network available in MATLAB environment are GoogLeNet, AlexNet.

Rovetta A.y. 2020/2021(a) Rovetta A.y. 2020/2021(b)

Chapter 2

Lab 5a: Autoencoder

To execute the first part of the lab, consider as input the MNIST dataset, composed of images handwritten digits from 0 to 9. Considering the large dataset, we had to select a subset composed of only two digits with reduced dimension (500 randomly samples for each). The experiment is done with different combinations (some will be easier to learn (for instance 1 and 8), some harder, (for instance 1 and 7). To train and encode the dataset, two made Matlab funcions are used: "trainAutoencoder()" and "encode()"; The num of units for the hidden layer is 2, in such a way we can have 2-dimensional plot. The output is the value of its hidden layer: we hope that similar patterns will have similar representations. To plot the results, we had to use the given "plotcl()" function.

2.1 Results

The evaluation of the task was done for different combinations of two digits, considering 2 units of the hidden layer: indeed in the following plots, we have got two different colors point which represent two neurons.

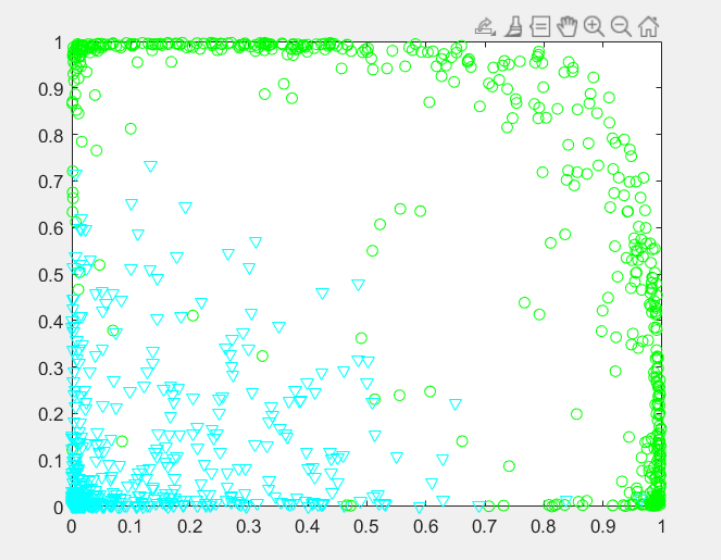

Figure 2.1: Digits 1 and 8

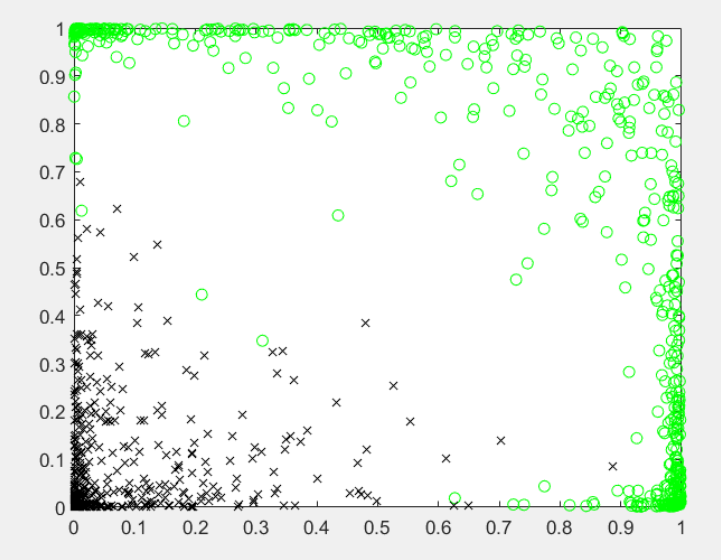

Figure 2.2: Digits 1 and 4

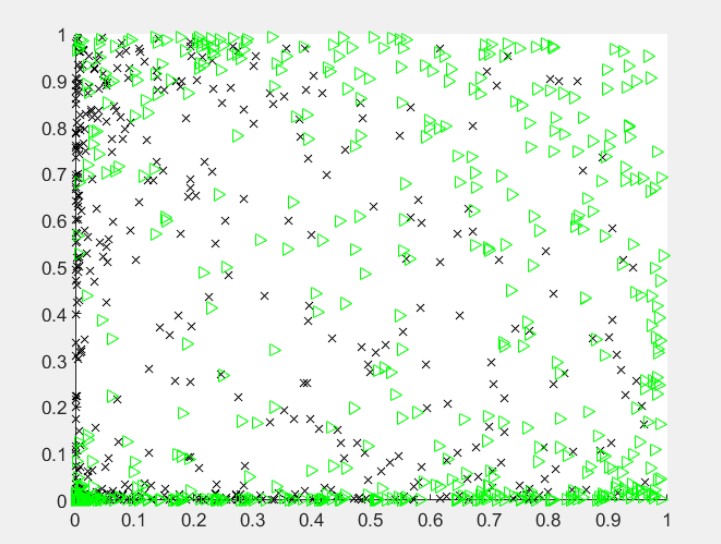

Figure 2.3: Digits 4 and 9

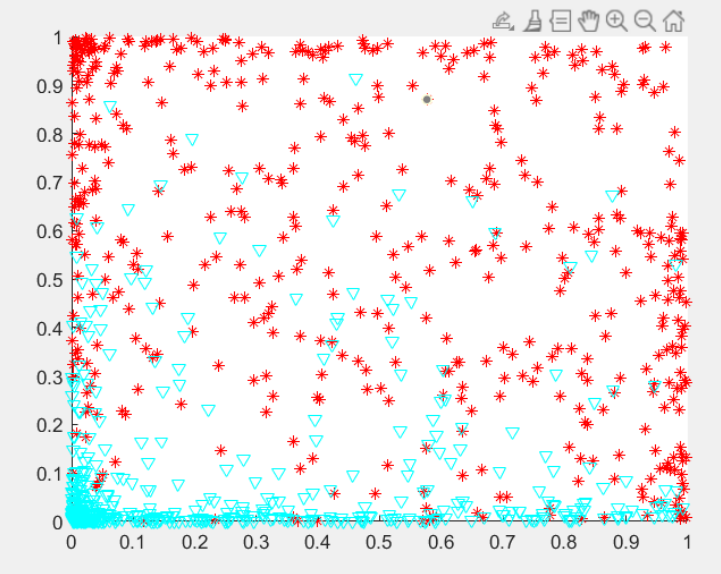

Figure 2.4: Digits 3 and 8

As we can see, if the shapes of two digits are very different to each other, then the different points are easily linearly separable (for instance 1-8 and 1-4), otherwise, the points are mixed, and to separate them results hard (for instance 4-9 and 3-8).

Applied Deep Learning - Part 3: Autoencoders 2017

Chapter 3

Lab 5b: practice with deep learning

The goal of this lab, as said, is based on detection and classification of images. The lab is composed of 3 tasks:

1) Task 0: Deep Learning Matlab demo, in which we are going to get acquainted with the programming style of MATLAB's deep neural networks.

2) Task 1: Use pretrained Convolutional Neural Networks (GoogLeNet) for image classification.

3) Task 2: Use pretrained Convolutional Neural Networks (ResNet-18) for Feature Extraction.

4) Task 3: Use pretrained Conolutional Neural Network (GoogLeNet) for Transfer Learning.

3.0.1 Task 0

A Matlab tutorial is provided, to get practice with a deep neural network. The basic concepts for many machine leaning and neural network libraries are the same.

The webcam is used as camera to acquire input, and the CNN used is AlexNet. The network is capable of to classify images given a labels to them. Therefore we needed to install the MATLAB support package for USB Webcams and the toolbox model for AlexNet Network.

Results

Figure 3.1: Mouse - Task 0

Figure 3.2: Microwave - Task 0

The network is able to manage sharp pictures, and to classify them giving a label.

3.0.2 Task 1

In the task 1, I have followed the tutorial on how to classify an image using the pretrained deep convolutional neural network GoogLeNet. The network takes as input an image and gives as output the label for the object and the probabilities for each of the object categories. Moreover, the network can give as output the top 5 predictions for the image based on the probability of each classification as well. The image used as input have been taken online.

Results

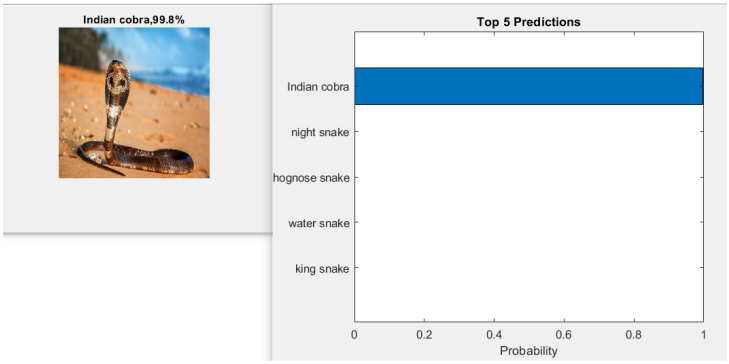

Figure 3.3: Cobra - Task 1

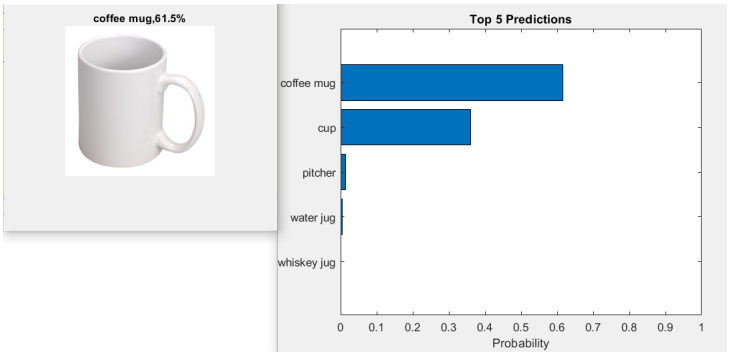

Figure 3.4: Tazza - Task 1

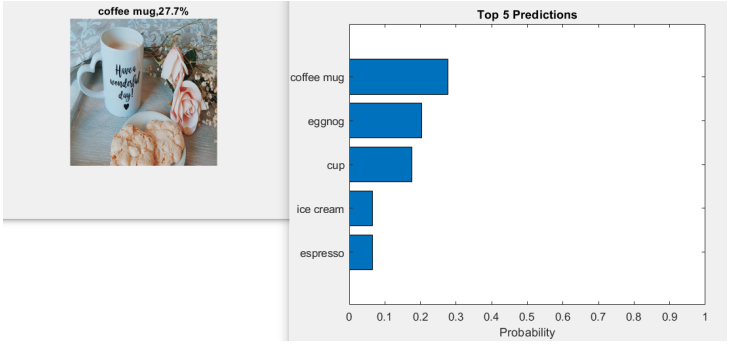

Figure 3.5: Tazza2 - Task 1

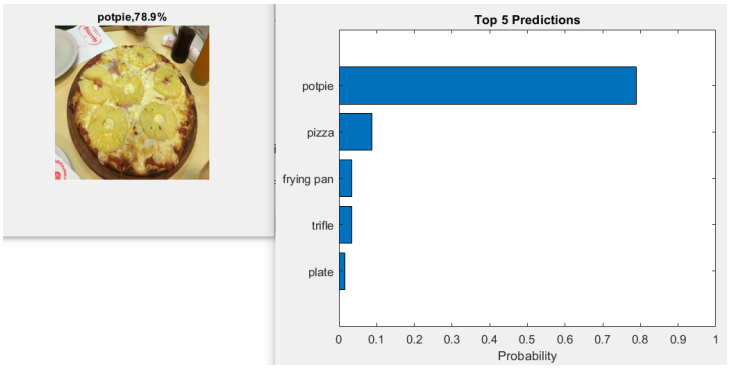

Figure 3.6: Pizza - Task 1

As we can see, some examples have been taken into account, and even if the object is not centralized on the picture and it is not sharp (for instance the figure 3.5) , the network is resulted able to classify it.

An important result is the last test (figure 3.6), in which the network was not able to classify the picture. The object is a kind of pizza with pineapple on the top. According to the network, it has not been classified as pizza, and in fact it is not!

3.0.3 Task 2

In the task 2, a pretrained network (ResNet18) has been used similarly to what I did with the autoencoder: it is implied as a feature extractor by using the layer activations as features.

Feature extraction is an easy and fast way to use the power of deep learning without investing time and effort into training a full network. You extract learned image features using a pretrained network, and then use those features to train a classifier, such as a support vector machine (SVM).

ResNet18 presents a total of 71 layers, the activation function is used at the level of the poor15 layer.

The dataset used is the "MerchData".

Results

Figure 3.7: Result

The figure 3.7 shows the classification of four different random images on a total of 50 taken from the "MerchData". In this case the accuracy is 1. If, for instance, a bigger dataset has been taken, than the accuracy could decrease.

3.0.4 Task 3

In the task 3, I have learned how to use an existing network as the initialization for a new training. Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.

It is useful when the dataset that has been taken into account is very small.

The pretrained neural network was GoogLeNet.

Optionally, it is possible to "freeze" the weights of earlier layers in the network by setting the learning rates in those layers to zero. During training, "trainNetwork()" does not update the parameters of the frozen layers. Because the gradients of the frozen layers do not need to be computed, freezing the weights of many initial layers can significantly speed up network training. If the new data set is small, then freezing earlier network layers can also prevent those layers from overfitting to the new data set.

Results

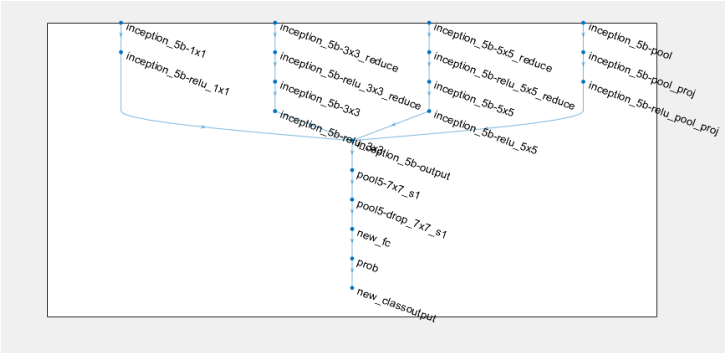

Figure 3.8: New Layers Graph for GoogLeNet

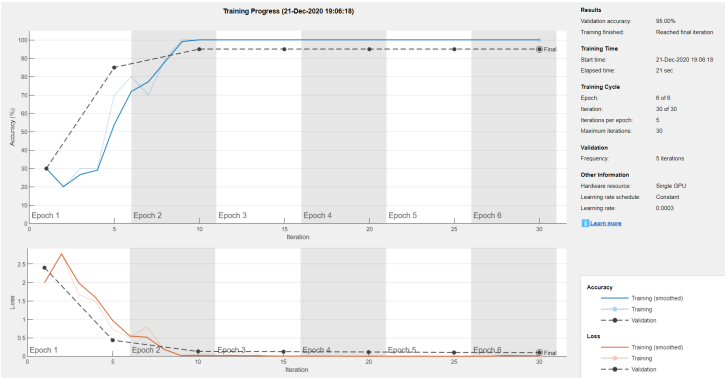

Figure 3.9: Training Progress

Figure 3.10: Example of classification

Deep Learning 2020 *Major Architectures of Deep Networks* 2016 *Deep Learning* 2020 *Convolutional neural network* 2020 *Transfer learning* 2020 *Try Deep Learning in 10 Lines of MATLAB*

Code 2020 *Classify Image Using GoogLeNet* 2020 *Extract Image Features Using Pretrained Network* 2020 *Train Deep Learning Network to Classify New Images* 2020

References

Applied Deep Learning - Part 3: Autoencoders (2017). Towards data science. URL: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798.

Classify Image Using GoogLeNet (2020). Mathworks. URL: https://it.mathworks.com/help/deeplearning/ug/classify-image-using-googlenet.html.

Convolutional neural network (2020). Wikipedia. URL: https://en.wikipedia.org/wiki/Convolutional_neural_network.

Deep Learning (2020). Wikipedia. URL: https://en.wikipedia.org/wiki/Deep_learning.

Extract Image Features Using Pretrained Network (2020). Mathworks. URL: https://it.mathworks.com/help/deeplearning/ug/extract-image-features-using-pretrained-network.html.

Major Architectures of Deep Networks (2016). O'Reilly. URL: https://www.oreilly.com/library/view/deep-learning/9781491924570/ch04.html.

Rovetta, Stefano (A.y. 2020/2021[a]). *ml-2020-21-08*. URL: https://2020.aulaweb.unige.it/pluginfile.php/276344/mod_folder/content/0/ml-2020-21--08.pdf?forcedownload=1.

– (A.y. 2020/2021[b]). *ml-2020-21-08*. URL: https://2020.aulaweb.unige.it/pluginfile.php/276344/mod_folder/content/0/ml-2020-21--09.pdf?forcedownload=1.

Train Deep Learning Network to Classify New Images (2020). Mathworks. URL: https://it.mathworks.com/help/deeplearning/ug/train-deep-learning-network-to-classify-new-images.html.

Transfer learning (2020). Wikipedia. URL: https://en.wikipedia.org/wiki/Transfer_learning.

Try Deep Learning in 10 Lines of MATLAB Code (2020). Mathworks. URL: https://it.mathworks.com/help/deeplearning/gs/try-deep-learning-in-10-lines-of-matlab-code.html.