Convolutional Neural Networks for object detection in professional appliances

Laura Meneghetti,* Nicola Demo*, Gianluigi Rozza* and Mauro Sonego**

*Mathematics Area, mathLab, SISSA, International School of Advanced Studies, Trieste, Italy

**Electrolux Professional, AD&T, The Research Hub, Pordenone, Italy

Introduction

We present a challenging problem as object detection and its application inside a leading company in the field of professional appliances, such as Electrolux Professional. Starting from the introduction of the basic notions about Artificial Neural Networks, we then describe how to deal with the problem of Image Recognition using Convolutional Neural Networks (CNNs) and how to extend CNNs in order to solve the task of Object Detection. We then explain how we have implemented everything using PyTorch and the results obtained.

1 - Artificial Neural Networks (ANNs)

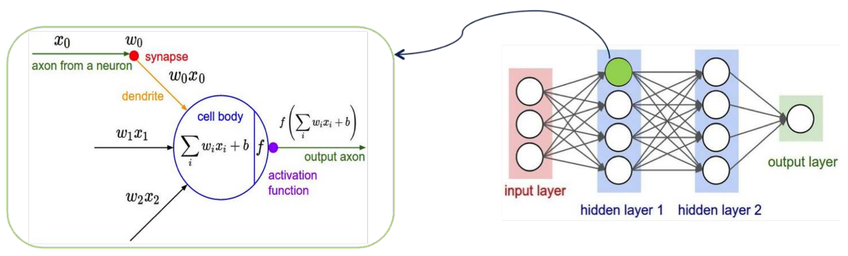
Artficial Neural Networks (ANN) are brain-inspired systems which are intended to replicate the way that we humans learn.
Given an ANN with L layers and $x = (x_1, \dots, x_N)^T$ an input vector. The output of a neuron in the ℓ layer is given by

$$y_j^\ell = f\Big(\sum_{i=1}^{N} w_{ij}^\ell y_i^{\ell-1} + b\Big)$$

- $w^\ell = (w_{1,1}^\ell, \dots, w_{N,1}^\ell)^T$ vector of *weights*
- f *activation function* (e.g. sigmoid, tanh, ReLU)
- $y_i^{\ell-1} = x$ for $l = 1$, i.e. when we are considering the first layer.

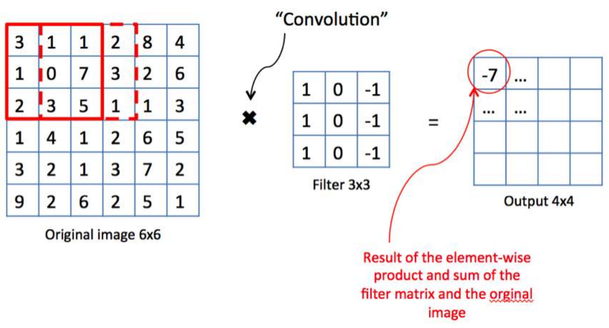
- Forward-Propagation: $\longrightarrow$ Testing Phase (Prediction)
- Back-Propagation: $\longrightarrow$ Training Phase (Learning)
  $\rightarrow$ Need to minimize a **Loss function** $L(y, f(x; W))$ using a Gradient Descent tecnique.

$$\Delta W = \frac{\partial L(y, f(x, W))}{\partial W} \qquad W \leftarrow W - \alpha(\Delta W + \lambda W)$$



2A - Convolutional Neural Networks (CNNs)

Deep Learning algorithms able to assign importance to various aspects/objects in an image, given as input, and to differentiate one from the other.
$\longrightarrow$ Convolutional Layers that uses filters to extract feature maps



$\longrightarrow$ Solve problem **Image Recognition**
Example of CNNs: VGG16, ResNet, GoogleNet, ...

2B - Object Detection

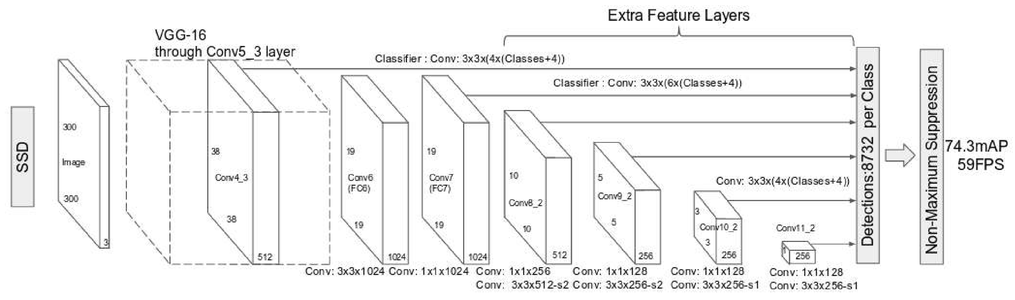
Goal: Classification + Localization of objects in pictures

- Classification part $\longrightarrow$ CNN: extract the low-level features
- Localization part $\longrightarrow$ Need Extra Features Layers:
  - Convolutional layers to extract high level feature maps
  - Convolutional layers to locate and identify objects in these feature maps.

Two different types of Object Detectors:

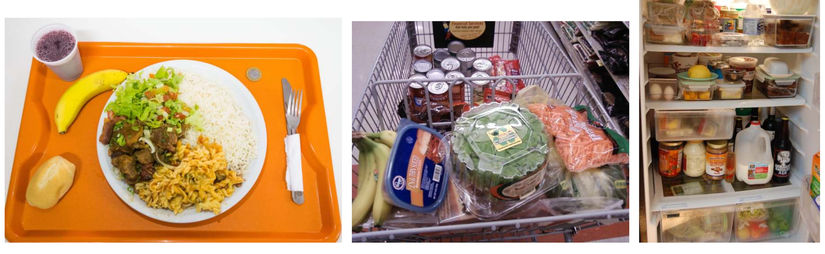
1. **Two-Stage Detectors**: Solve the two tasks of object detection in two separate steps: first you find an arbitrary number of objects using a region proposal network. $\longrightarrow$ Faster R-CNN
2. **One-Stage Detectors**: The two tasks are done in a "single shot", simultaneously predicting the bounding box and the class as the image is processed. $\longrightarrow$ SSD

Example: SSD



3A - How to implement your own Object Detector

1. **Dataset**: Images containing objects of different categories, such as different types of food or beverages, ... $\longrightarrow$ split for training (20%) and testing (80%)



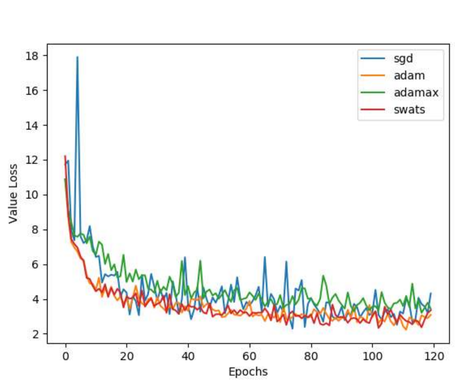
2. **Preparation of the dataset**: use *LabelImg* app to label the images and create bounding boxes using Pascal VOC notation.
3. **Implementation**: creation of our own Python library for Object Detection.
   Architecture :SSD300 using VGG-16 pretrained on ImageNet as base newtork.
   Programming Language: Python 3.7.5 using PyTorch 1.3.1

3B - Future Work

- Analyze ROM-type approaches for CNNs
  $\rightarrow$Accelerate the training process
  $\rightarrow$ Reduce memory consumption for grafting on embedded systems.
- Comparison of results using another architecture as Faster R-CNN.
- Understand how to better adapt the learning rate in SGD optimizer.

4 - Results

Training 120 epochs using different optimizers: SGD, Adam. Adamax, Swats. Needed ~ 5 hours to run on CPU.



References

[1] T. Daulbaev, J. Gusak, E. Ponomarev, A. Cichocki, and I. V. Oseledets. Reduced-order modeling of deep neural networks. *CoRR*, abs/1910.06995, 2019.

[2] I. J. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.