Università
degli Studi
di Padova

DEPARTMENT OF
INFORMATION
ENGINEERING
UNIVERSITY OF PADOVA

Retail Shelf Analytics Through Image Processing and Deep Learning

Master Thesis

Department of Information Engineering
University of Padua, Italy

Student
Alvise De Biasio

Supervisor
Carlo Fantozzi

February 2019

A.Y. 2018/2019

Dedicated to my girlfriend, my family and my friends.
Best ideas shine with the passion of their creators.

Preface

This work is for people who love reasoning and who like to discover how things work.

When we dive deep into the deep learning world we are immediately fascinated by the continuous research of the first principle, the triggering factor, the cutting-edge technique and above all the elegant rules that govern complex systems. We could find these themes that appear technical and sophisticated in many fields of our daily life. In particular, AI and deep learning are still the symbol of a revolution that accelerates the innovative development of different architectures such as industrial automation systems, medical devices, chatbots and enterprise management systems.

The present thesis aims to promote an innovative approach based on modern deep learning and image processing techniques for retail shelf analytics within an actual business context. To achieve this goal, the research focused on recent developments in computer vision while maintaining a business-oriented approach. In particular, some of the themes that will be discussed in the following chapters that could be considered the most valuable and the most expensive part of this work are respectively the construction of a dataset able to adapt to the Customer's business domain, the study of the algorithmic component and the business integration project between the Customer and Supplier IT systems. The project involved the full-stack software development of a SAP-based B2B product to analyze structured and unstructured data and provide business intelligence services for retail products analytics. This project comes from the collaboration of three different parties that have contributed to its success. The research and business world met respectively in the robes of the University of Padua and the Supplier and Customer companies. During the entire project, interaction between the parties was constant in order to integrate the R&D work with the Customer's processes. Particular attention was given to the development of a deep learning architecture based on TensorFlow able to work at scale. The results from the deep learning software are based on a system integration project between IT Customer and Supplier systems. The software developed in this project has been purchased by the Customer and will be further developed in the following months. Future work will be based on recent scientific advancements in deep learning and will focus on the development of the Customer's business growth strategy. To achieve this goal, we will study the integration of new value-drivers into the current system that will allow the Customer to have a competitive advantage in the market. First of all, KPIs will be defined to evaluate the sales performance of individual products on different customers target groups. Based on the defined KPIs and the sales forecasts of the different products, a recommendation system will be studied to suggest promotions and discounts to the final customers. More in-depth considerations on these themes will be described in the conclusion chapter.

Acknowledgements

A special thanks to all those who will venture in the reading of this thesis, I hope this product can thrill you and provide new key insights.

First of all, I would like to thank Prof. C. Fantozzi for giving me the opportunity to develop this work. Deep expert of the subject and patient reference throughout the duration of the journey, he gave me valuable suggestions stimulating a natural attitude to curiosity. I really appreciated your help and your support in the various phases of this project. Thank you very much for the constructive comparisons, the key insights and the ideas provided to this work, as well as for the availability shown throughout this thesis period. It was a pleasure and an honor for me to have had the opportunity to collaborate in this work.

Afterwards, I am pleased to thank Dr. A. Bergamo, Dr. S. Citron, Dr. P. Trevisan and the entire Estilos srl team. An engaging international valuable team that has been able to coordinate the interactions with the Customer and has provided methods and techniques in an enterprise scenario. In particular, a grateful thanks goes to Dr. A. Montagna with whom I have been able to get in touch on the most interesting themes in the business logic world.

I would also like to thank all the people with whom I have been able to interact during this period, for the encouragement and for contributing positively to this work. Especially i would like to thank my girlfriend Anna for being the most patient, kind and loving person with whom I have been in contact during these years. A heartfelt thanks for being with me in happy and difficult times. I thank you for your honesty, your courage and the love you put into everything you do. I am also delighted to thank Prof. S. Ghidoni for being a support and a friend in my master's degree studies. During my bachelor's degree he disrupted my previous vision and changed my way, getting me closer to the computer industry. Finally, my biggest thanks go to my family, especially to my mother, my friends and all the people dearest to me. Thank you for supporting me every day: part of this work is also yours.

Contents

Abstract

Recently, deep learning has emerged as a disruptive growth potential for the development of new business domains. Modern AI techniques are used in the most innovative projects of the largest market players in all the industrial fields. In particular, automated retail analytics applications are still integrated in enterprise solutions targeted towards a specific business audience.

This work represents a concrete case where the business world meets the research world. In particular, the project has been developed in collaboration with the University of Padua, for a Customer company that plays a role of strategic importance in the international pharmaceutical retail market. The themes that will frequently occur within this work and which they have constituted the most valuable and the most expensive part are respectively the creation of a dataset able to adapt to the Customer's business domain and the business integration project between the Customer and Supplier IT systems. As for the dataset, a lot of time has been invested to plan every single feature in detail, respecting the requirements defined in the design phase. In particular, in addition to the manual pixel-wise labeling of all the images, design activities have been followed regarding the selection of the tools for the construction of the same as well as planned experimental tests to investigate the effects of automatic labeling on the final deep learning predictions. Again with regard to the latter topic, a deep learning pipeline has been studied to integrate the traditional inference process in order to obtain accurate predictions without the need to own large amounts of data for the training of the neural networks stack.

Regarding the business integration project, the work has been sold through a Supplier company with a SAP based B2B product. The design involved the full stack development of an innovative software solution for the automated analysis of shelf products in store images based on deep learning and image processing techniques. According to the specifications, the system model aims to analyze structured and unstructured data to provide key insights on the Customer brand impact over final sales points. Development proceeded with the design of a convolutional networks stack to detect and segment different product instances in store shelf pictures. The proposed approach evaluated different alternatives that make a more or less intensive use of structured manual annotations. The prototyping exploited an experimental evaluation system for performance benchmarking to fine-tune the network architecture according to the studied business domain. In this context, several tests investigated the final system trade-off between precision and recall to reach the best results for the Customer activities. The current system is able to provide processing at scale by integrating directly into a standardized data flow between the Customer and Supplier systems. The results can be consulted directly in the cloud through a SAP-based presentation logic layer. The automated reporting provides a series of KPIs that assess the effectiveness of the Customer brand on the market. This business intelligence information could be used directly by the Customer to gain competitive advantage over its business competitors. Future work will see a more targeted integration within the Customer processes to evaluate the correlation of the stands with the final sales performance of the individual products. A recommendation application will be studied to suggest the optimal products exposure on store shelves based on sales forecasts. Other value-drivers concerning the development of the Customer business which will be integrated directly into the current system, will be presented in the conclusion chapter.

Chapter 1

Introduction

In recent decades, technology has assumed an increasingly important role in daily life. The scaling of electronic components has made it possible to use hardware infrastructures that provide complex results quickly and at affordable costs. New performing systems have been introduced on the market, based on AI and deep learning for image processing, that have been able to draw from a mature and multidisciplinary articulated environment such as computer vision.

Being able to identify with a few words the main domains of AI and computer vision is complicated because the fields of interest are many and in continuous expansion. In particular, they range from pure entertainment to industrial robotics and enterprise systems exploring issues in security, medicine, architecture, art and education. These themes are just some of the most popular ones, but there are many hybrids born through a combination of the above. Moreover, what surprises every time is the fact that every innovation inevitably affects at least one other discipline. So it happens, for example, that research and development groups create such sophisticated algorithms to give new life to the world of retail or enterprise management systems. This could in turn renew the rules of video surveillance, or create new innovative techniques for medical images analytics through an absolutely dynamic and unexpected process [69, 39]. This happens because each technique is increasingly interconnected and innovation reverberates over adjacent business areas. The benefits deriving from such advanced technologies are evident. The constancy, reliability and objectivity of the technological architectures facilitate human intervention in processes that require multiple repetitive control operations. In particular, automated retail systems have recently reached very high levels of accuracy and practicality, managing to manipulate a growing number of information [26, 69]. This continuous development has allowed to explore ever wider fields, mitigating the problems of scalability and technological efficiency from time to time.

This thesis stands as a contact point between business and research. In particular, in the following chapters a concrete case of implementation of a retail shelf analytics system will be discussed (Chapters 4 and 5). The system has been sold to – and is currently used by – an important player in the international retail pharmaceutical market, from now on referred to as the Customer. The project is part of a context of innovative improvement of the Customer business processes. The development has been followed in collaboration with the University of Padua by a consulting company referred from now as the Supplier.

The parts of this work that make up the most valuable and the most expensive component are respectively the instance segmentation dataset creation to train the deep learning model based on Mask R-CNN [30] at the base of the elaborations of the system model and the business integration project to supplement the IT Customer and Supplier systems. The design of the dataset (Section 4.5) has been one of the most precious investments of this work. In particular, an instance segmentation dataset has been built targeted on Customer business processes that can be used to train deep learning models for pixel-wise predictions of Customer and Competitors products on store shelves images. The activity has been necessary because as highlighted in Section 1.1 there are currently no publicly released datasets able to obtain satisfactory performance on the business domain considered. Along with the manual labeling activity for every single pixel of all the images of the dataset, other design activities have been followed regarding the development of experimental tests to assess the effects of automatic labeling tecniques on the final deep learning predictions (Subsection 5.2.2). Regarding the latter topic, as described in Section 4.4, a deep learning pipeline has also been studied to abstract the traditional inference chain in order to obtain accurate brand-level predictions without the need to own large amounts of data for the training of the Mask R-CNN neural networks stack.

As far as the business integration project (Section 4.4) is concerned, all the design phases of a full-stack R&D solution that integrates the traditional Customer's business processes have been taken care of. As described in Chapter 4, the data logic, the business logic and the presentation logic have been designed in detail in compliance with the SAP architecture agreed in the requirements analysis (Section 4.1). All the consulting activities with the various Customer departments have also been followed. In compliance with the confidentiality constraints imposed by Customer and Supplier, some of the most important details have been obscured to maintain anonymity where required. However, as for the construction of the dataset, this activity has also been a very valuable investment for the success of this work.

The designed system is able to process images of in store shelves products at scales and integrates directly with the IT Customer and Supplier systems through a dedicated data flow. The work takes advantage of the modern instance segmentation techniques [20, 30] that will be used to identify and segment the different instances of shelf products exposed in pharmacies. These techniques will be described in detail in Chapters 3 and 4 respectively and represent the core processing of the system model that aims to extract metrics and KPIs of strategic importance for the Customer. As described in Chapter 4, this approach has been preferred to the more common object detection approach by the Supplier in the requirements analysis phase. In particular, the Mask R-CNN algorithm [30] has been chosen for the possibility of having pixel-wise predictions and consequently to calculate with greater precision the exposed surface for the various products in the pictures. As a result of the algorithm elaborations, from the images a complete series of key insights is extracted that bind to the Customer business processes to investigate the visibility of the Customer brand in retail stores (Section 5.3). The results can be consulted directly in the cloud within an automated business intelligence reporting in a SAP-based B2B product.

The rest of the thesis is organized as follows. In the present chapter we illustrate the state of the art of modern shelf product recognition systems (Section 1.1). The main techniques, and the datasets available with an open license, are described. An Ad-Hoc dataset, built during the work for this thesis and addressing the issues of available datasets, is presented. Then, Chapters 2 and 3 are dedicated, respectively, to a short survey on deep learning and to the state of the art of modern object detection algorithms. The former chapter is a self-consistent kernel for the understanding of modern architectures based on neural networks, and in particular on convolutional neural networks. The latter is a path that discusses the recent discoveries in the object detection field by presenting the main strategies from an evolutionary point of view. Chapters 4 and 5 are dedicated to the detailed description of the system model we developed, and to the main results achieved. The various development phases are described in detail and particular care is taken in the description of the deep learning algorithms used and in the integration of the latter with the IT Customer and Supplier systems. Finally, the main results and key insights from the experimental tests that explore the optimal configuration of the network architecture stack as well as the business intelligence and visual analytics results on the Customer processes will be presented.

1.1 Related Work

Retail is a very profitable business that develops many innovative themes in the 4.0 industry field with specialized commercial products or automated retail infrastructures such as Amazon Go, a partially-automated grocery stores chain where customers can buy products without having to go through any kind of checkout station [26]. Amazon Go exploits advanced computer vision and deep learning techniques combined with specific sensors to automate the purchase of in-store products.

However, even if there are many commercial products, at present retail has not been studied

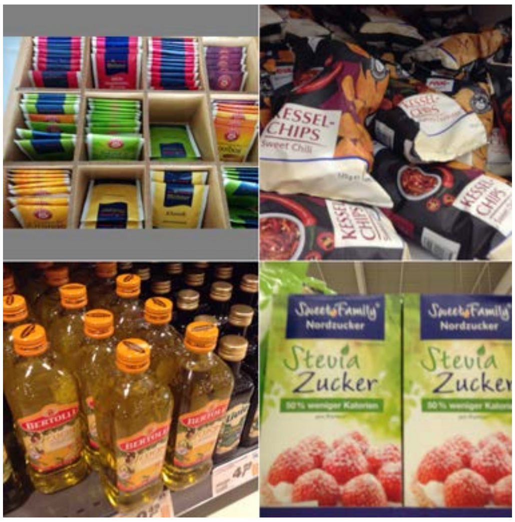


Figure 1.1: Image classification of in-store products – Freiburg Groceries Dataset [38].

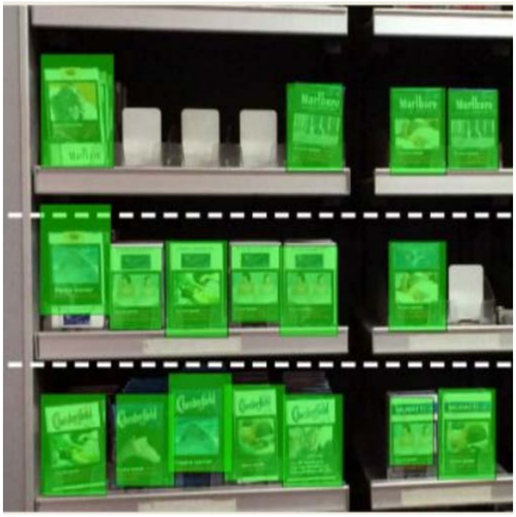


Figure 1.2: Object detection of products on store shelves – Tobacco Dataset [75].

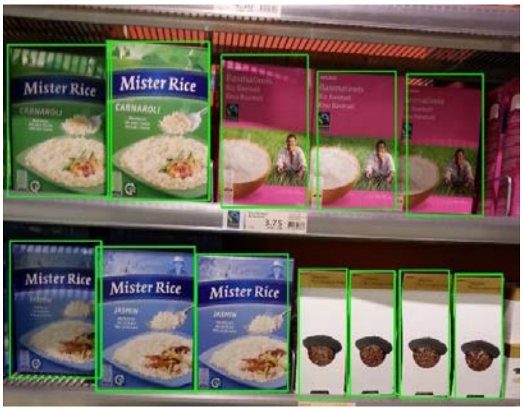


Figure 1.3: Object detection of products on store shelves – Groceries Products Dataset [71].

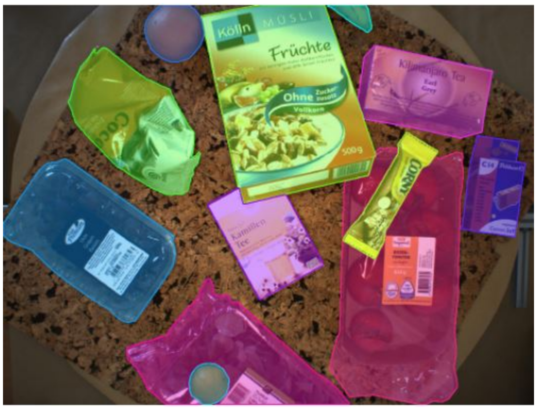


Figure 1.4: Automatic product checkout – Densely Segmented Supermarket Dataset [17].

yet in detail in its declinations. In this context, very few scientific papers exploit computer vision techniques to recognize in-store products. This is mainly due to the fact that:

- the largest and most studied computer vision datasets such as Ms Coco [46] and ImageNet [40] do not provide annotations to tackle the in-store product recognition problem;
- there is a very large context and product variability, and a huge effort is required to manually annotate the individual products in any images taken in-store;

Given the previous difficulties, the main works try to study the problem from different perspectives. In particular, according to the requirements, specialized datasets are constructed but often these are not very extensive or even publicly available.

In 2014, M. George [21] proposed an approach for per-exemplar object detection of products in store images. The approach showed promising results. At the same time, the reference dataset (Groceries Products Dataset), containing 8350 different images of single products on a white background and 885 in-store test images, was made available. In 2015, G. Varol [75] studied the problem of recognizing products on shelves through a different approach. In particular, an object

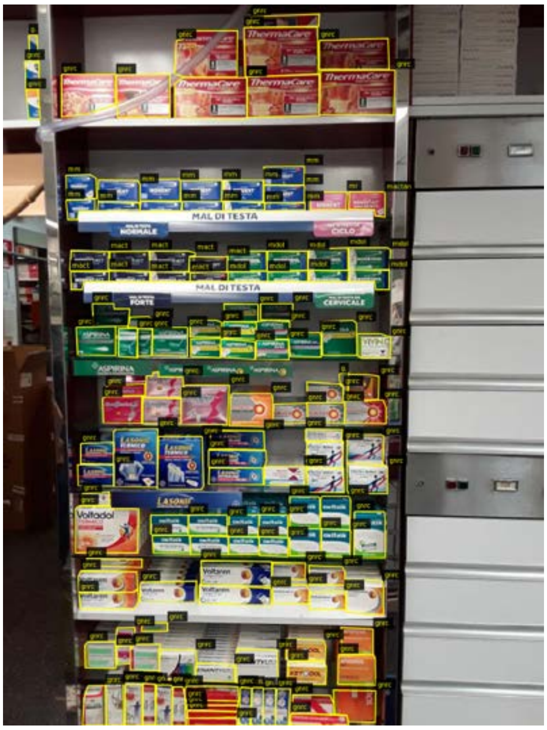


Figure 1.5: Test Set annotations – Ad-Hoc.

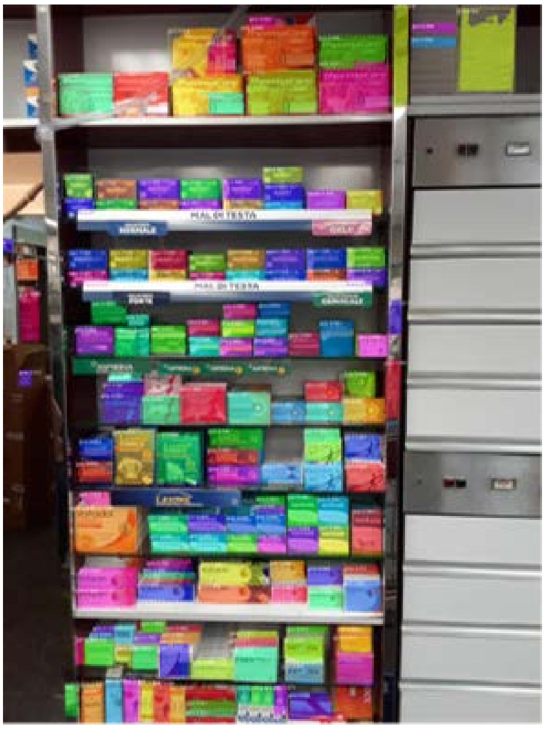


Figure 1.6: Test Set inferences – Ad-Hoc.

detection system based on HOG [14] and Viola-Jones [76] was proposed to recognize products and shelves. The same system succeeded in further specializing the detection of individual products at the brand level through a bag of features approach [41] based on SIFT features [49] for shape description and HSV values for color description. Even in this case the dataset, based on 354 tobacco images acquired from 40 groceries, was made public. The problem of in-store recognition has been further studied by Jund in 2016 [38] with an image classification approach of in-store images based on convolutional neural networks. The study has open-sourced an additional dataset for image classification (Freiburg Groceries Dataset) that consists of 5000 small images covering 25 different food classes. The recognition of products on store shelves was taken up again in 2016 through a logo recognition approach based on non-deep detectors [54].

Two further studies concerning the recognition of products in store shelves were recently carried out by A. Tonioni et al. in 2017 and 2018. The first study [70] investigates the problem through a non-deep approach based on the previously published Groceries Product Dataset [21]. This is due to the impossibility of using region-based or single-stage object detection approaches on the annotations in the dataset. The second study [71], instead, proposes a deep learning object detection pipeline based on YOLO [58, 56, 57] and the KNN algorithm [12] to differentiate both the product class and the associated brand. However, although in this case the test set is available for evaluations, the training set used for the YOLO detector has not been disclosed.

In 2018, further studies have been carried out in the context of weak supervision and automatic checkout. In the first study [74], a weakly supervised approach is used to identify shelf products using synthetic datasets. In the second one [17], an instance segmentation approach is proposed for the recognition of products in order to automate checkout systems. The latter work has produced an open source a dataset (Densely Segmented Supermarket Dataset) containing 21000 high-resolution images of products on check-out desks labeled pixel-wise from 60 different product categories.

From the analysis of the studies in the literature, there is currently no dataset that can be adopted for the recognition of products on store shelves with instance segmentation [30, 20] techniques. As described in Chapter 4, this approach has been preferred to the more common object detection

Name	Description	Size	Class	Tasks	Format
Groceries Products	Per-exemplar object detection	9K	8.3K	Image classification, object detection	RGB
Tobacco Groceries	Object detection on store shelves	354	10	Object detection	RGB
Freiburg Groceries	Small images, various classes	5K	25	Image classification	RGB
D2S	Automatic checkout	21K	60	Object detection, instance segmentation	RGB
Ad-Hoc	Instance segmentation on store shelves	410	8	Object detection, instance segmentation	RGB

Table 1.1: Main deep learning object detection open-source retail datasets [38, 75, 71, 17].

approach by the Supplier and the Customer in the requirements analysis phase (Section 4.1). In particular, the Mask R-CNN algorithm [30] has been chosen for the possibility to calculate with greater precision the exposed surface for the various products in the pictures. This is due in particular to the characteristics of the algorithm to make pixel-wise predictions for each object identified in the images (Subsection 3.2.5). The system model described below aims to address this survey dimension and implement an instance segmentation architecture of shelf products using advanced deep learning techniques based on the Mask R-CNN algorithm. Given the impossibility of finding a suitable dataset, an ad-hoc one has been designed based on images provided by the Customer (Section 4.5). The dataset extends the landscape of the possible applications of the algorithms of object detection and instance segmentation and as discussed in Chapter 5, good results have been obtained in the inferences of all its classes. This consists of 410 segmented pixel-wise high-resolution images for a total of 8 categories. For model selection purposes this dataset has been further divided into a 300-image training set and a test set with 110 images. The dataset has proved to be robust to use in the business domain under investigation and is currently the most important resource for training the network architecture stack at the base of the designed system model. The importance and the key features that this dataset offer, as well as the main challenges it implies, will be reported in Chapters 4 and 5. In particular, while the first will give a design overview, the second will present some experimental tests. With respect to the last two points, in addition to the manual labeling of every pixel of all the images many of the efforts were made in attempts to automate the dataset labeling process as well as in designing a pipeline capable of recognizing shelf products without owning large amounts of data for the training of the neural networks. To defend the intellectual property of the Customer, this dataset will remain proprietary and will not be disclosed.

This work proposes an automated retail shelf analytics system based on the ad-hoc dataset described above. Two different alternatives will be presented that exploit a more or less intense use of manual annotations. This fact translates into a performance trade-off between level of detail and accuracy of the final inferences, which will be discussed in Chapter 5. Both architectures are based on the recent results of Mask R-CNN [30] and can recognize and segment the different instances of products in store images. The elaborations has been integrated directly into the business processes of the Customer and will allow to obtain key insights of strategic importance to evaluate the impact of the Customer brand in the final sale points (Section 5.3).

Chapter 2

A Deep Learning Survey

Machine learning, deep learning, predictive analytics, and AI in general are just some of the hot topics nowadays. The ongoing revolution follows a continuous process of disruptive growth and AI is a very important asset in today's enterprise landscape. From a management point of view, AI has radically transformed business processes, from transport to industries to communications to health care. In AI, deep learning is currently one of the main drivers of worldwide innovation.

This chapter offers a brief survey of deep learning. The basics of modern deep learning algorithms, with emphasis on modern convolutional neural networks and the technological aspects that led to the most recent discoveries, will be discussed.

2.1 Deep Learning Development Scenarios

Programmatically designed features are often incomplete, redundant, and a lot of time is spent in the underlying core logic design. Features derived from machine learning algorithms are generally quick to design and easily adapt to different contexts. Deep learning constitutes a scalable, flexible, universal and automatic mechanism to represent data relationships [55].

In particular, deep learning is a sub-field of machine learning. However, while machine learning tries to describe data with features that a machine can understand, deep learning tries to automatically learn good features and representations through sequences of features hierarchically organized on multiple levels [39].

Deep learning has no recent origins but it has acquired popularity just in recent years. In fact, only since 2010 results proposed by deep learning algorithms have started outperforming machine learning techniques [39, 55]. In particular, there are three main ingredients that have propelled the success of deep learning algorithms:

- new optimized algorithms have been designed,
- huge computational power has been achieved through technological progress,
- large amount of data have been available for models training.

These themes have been further developed and different entities collaborate daily in the design of increasingly optimized algorithms. There are several international challenges where algorithms are tested to improve over the state of the art [40]. There are technological implications to the point that entire industries design dedicated hardware and specialized software frameworks [51]. The phenomenon has radically changed the business world and currently deep learning is an hot topic with several practical applications.

Any type of data that is structured, unstructured or binary, may be the subject of deep learning algorithms. It is possible to analyze texts, images, sounds, sequences, and more. In particular, some of the most innovative themes try to investigate the relationships between visual and textual data. This is solved in multidisciplinary contexts such as computer vision and natural language

LEFT REARWARD VEHICLE CAMERA
MEDIUM RANGE VEHICLE CAMERA
RIGHT REARWARD VEHICLE CAMERA

Figure 2.1: Tesla self-driving car insights.

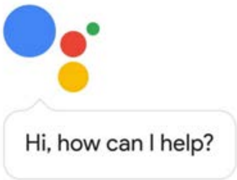


Figure 2.2: Google assistant.

processing for both research and enterprise applications. Self driving cars, robotics, sentiment analysis, chatbots and process automation are just some of the possible derivations of deep learning. Deep learning is the key to the most innovative projects of prestigious universities such as Stanford [39, 55] as well as well-known software industries such as Google [51] and many others.

2.2 Neural Networks Basics

A neural network is a statistical model that is traditionally used to solve engineering problems related to artificial intelligence. Currently, neural networks are a multidisciplinary research area that exploits different concepts related to mathematics, statistics, engineering, computer science, biology, psychology and other disciplines [39, 55].

According to biological interpretations, neural networks are computational models based on biological networks mechanisms and constitute a simplified model of the brain. Specifically, neural networks are adaptive systems based on interconnections of artificial neurons. Such interconnections are capable of modifying their structure based on both internal and external data.

The working principle of neural networks is the exchange of information between elementary computational units called artificial neurons. An artificial neuron is a mathematical model proposed by McCulloch and Pitts in 1943 [52] to simulate the logical mechanism of a biological neuron. The artificial neuron receives one or more inputs representing post-synaptic potentials at neural dendrites and sums them to produce an output, representing the neuron's action potential which is transmitted along its axon.

The Perceptron model, proposed by Frank Rosenblatt in 1958 [60] and the subsequent refinements proposed by Minsky and Papert in 1969 [53], constitute a more general computational model than McCulloch-Pitts neuron. In particular, the Perceptron overcomes some of the limitations related to the previous model by introducing the concept of numerical input weights and a mechanism for learning those weights. In 1959, the Mark I Perceptron [28], that currently resides in the Smithsonian Institute, was the first hardware implementation of the model. This allowed the study of the effects of combinations of input features from the experimental point of view.

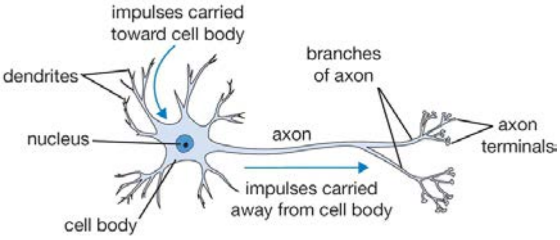

impulses carried toward cell body
dendrites
branches of axon
axon terminals
nucleus
axon
impulses carried away from cell body
cell body

Figure 2.3: Biological neuron schema [39].

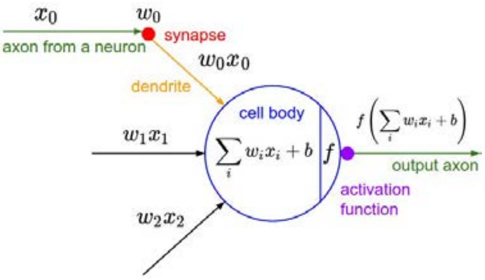


Figure 2.4: Artificial neuron schema [39].

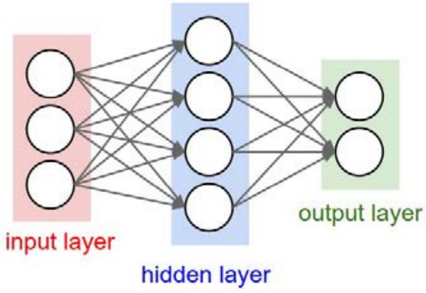


Figure 2.5: A 2-layer feedforward neural network with 2 output nodes [39].

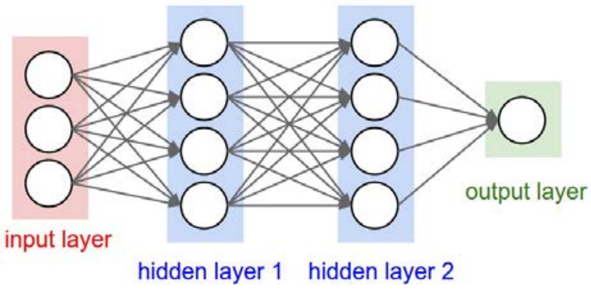


Figure 2.6: A 3-layer feedforward neural network with a single output node [39].

The mathematical form of the model neuron's forward computation has the capacity to activate or deactivate certain linear regions of its input space. It follows that, by combining the output of the neuron with a specific loss function, it is possible to obtain a linear binary classifier. In this context, the Perceptron learning algorithm constitutes an iterative algorithm that allows to determine an approximate value of the optimal input weights [39]. In particular, the algorithm aims to find, in the case of linearly separable data, the hyperplane that is able to discriminate the two input classes. As a result of convergence, after an initial training phase the algorithm will be able to produce inferences on new data.

Subsequently, these basic concepts evolved into successive optimizations managing the multi-class implications and the neural interconnections in the multi-layer case under the concept of feedforward neural networks. Although the logical model of neural networks does not have recent origins, its practical applications started to bloom only in 1886 with the discovery of the back-propagation method [61]. In fact, although neural networks are models capable of achieving great abstraction, these have traditionally been strongly constrained by training times. In the context of learning, backpropagation is commonly used by the stochastic gradient descent optimization algorithm to adjust the weight of neurons by calculating the gradient of the loss function.

The increase in performance of these complex structures has grown over time reaching ever higher standards. The interest of the scientific and industrial community has led to solutions for real-world applications in scenarios that were once unimaginable, from computer vision to natural language processing and process automation.

2.2.1 Structure of Feedforward Neural Networks

From a mathematical point of view, the artificial neuron 2.4 is a function

$$\boldsymbol{x} \to \sigma(<\boldsymbol{w}, \boldsymbol{x}>) \text{ with } \boldsymbol{x} \in \mathbb{R}^d \text{ and } \sigma : \mathbb{R} \to \mathbb{R},$$

called *activation function* [39, 55]. There are several activation functions that take inspiration from both biological insights and optimization concepts of the gradient descent algorithm. In particular, although the sigmoid function has been used frequently in the past for its nice interpretation as a saturating firing rate for a neuron [39], it has three main problems in its concrete use: sigmoids saturate and kill gradients, sigmoid outputs are not zero-centered and the exponential function is computationally expensive. For these reasons, other solutions have been designed to mitigate the previous issues. In particular, multiple advantages have been obtained with the Rectified Linear Unit (ReLU) since: it does not saturate in the positive region, it was found to greatly accelerate the convergence of stochastic gradient descent algorithm compared to the sigmoid functions [40], it is very efficient from a computational standpoint and it represents a more biologically plausible model than sigmoid. Other improvements have been made in this area with the development of activation functions that try not to saturate even in the negative region as Leaky ReLu [50], ELU [10], Maxout [24]. The study of these functions is still a field of active research.

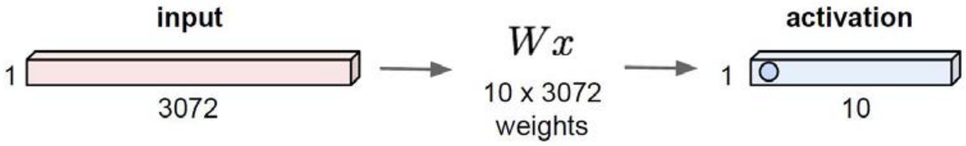


Figure 2.7: Fully connected layer [39].

From these elements it is possible to define a feedforward neural network [39] as a weighted directed acyclic graph $G = (V, E)$ organized into layers of artificial neurons where each edge $e \in E$ has a weight specified by $w : E \to \mathbb{R}$. The dimensions of modern neural networks are traditionally measured according to the number of layers and the number of learning parameters. Some of the most performing structures in this field are made up of dozens of layers and hundreds of millions of learning parameters. Computation is generally performed by repeated matrix multiplications interwoven with activation functions.

Fully Connected (FC) Layer

For traditional feedforward neural networks, one of the most common layer type is the fully connected layer where neurons of adjacent layers are fully pairwise connected and neurons within a specific layer share no connections [39]. So far there are many other types of layers that allow the network to have specific structural properties. For instance, convolutional layers are crucial in the computer vision field.

Given the architecture of a neural network specified by the tuple (V, E, σ) it is possible to define the representational power of a feedforward neural network composed by fully connected layers with the family of functions that are parametrized by the weights of the network. In particular, the results of Cybenko [13] demonstrate that neural networks with at least one hidden layer are universal approximators for the sigmoid activation function. However, although a single hidden layer is sufficient to approximate any continuous function, the trend in recent years has shifted to deeper networks. The shift is motivated by considerations of empirical nature and has not been demonstrated theoretically yet, which leaves many open possibilities for research.

These are just some of the insights behind the analysis of these complex structures, but many other problems have been addressed or are still subject of research [39, 55]. In particular, important aspects with open questions from both the experimental and theoretical point of view are the following.

Neural network architecture design. Common analytics themes concern the effects of layers architecture, network width/depth, weights initialization, regularization and ensemble learning on the network inferences. Moreover, recent studies investigate the effects of neural architecture search through reinforcement learning.

Training time optimization. Research focuses on optimization of stochastic gradient descent algorithm, effects of batch size and transfer learning [78].

Reducing the amount of training data. Some current development scenarios concern transfer learning, weak supervision approaches and data augmentation.

Interpretation of models. Some modern approaches consist in dimensionality reduction, neural activation visualization and neural style transfer.

Many of these themes will be treated during this dissertation, both from a theoretical and an implementation point of view. In particular, themes related to transfer learning [78] and optimization of the network structure [39] will be studied experimentally and will be documented through specific tests. Many topics will be presented as an introduction to more advanced concepts; for a more complete reference we refer the reader to [18, 39].

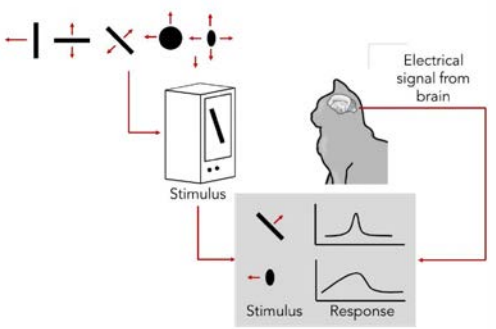


Figure 2.8: Hubel and Wiesel studies in cat's visual perception [34, 33, 39].

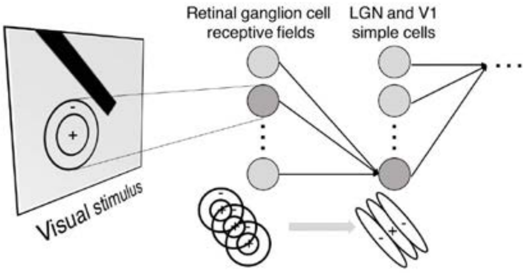


Figure 2.9: Hierarchical organization in visual perception [34, 33, 39].

2.3 Convolutional Neural Networks

Reading a road sign, recognizing a friend we meet in a cafe and even distinguish between different products at the supermarket: they all all seem easy tasks. However, if these are so simple for us it is only because our brains are incredibly good at understanding images. From a technical point of view the whole neural network architecture has made tremendous progress on addressing these difficult problems with a deep learning model called Convolutional Neural Network (CNN) [39].

From a historical point of view, in parallel with the studies that led to the characterization of some important properties of artificial neural networks, other studies have been carried out on brain cells that investigated similar mechanisms from the biological point of view. A series of studies from 1959 to 1962 were carried out by Hubel and Wiesel in order to determine the receptive fields, the binocular interaction and the functional architecture in cat's visual cortex [34] [33]. These studies, which provided for the stimulation of the visual cortex of cats and the measure of the response of brain activity in terms of electrical signals, have highlighted some particular properties.

Topographical mapping. Nearby cells in cortex represent nearby regions in the visual field.

Hierarchical organization. While simple cells response to light orientation, complex cells response to orientation and movement and hypercomplex cells response to movement with respect to an end point.

Starting from the intuitions derived from the studies of Hubel and Wiesel, novel neural networks were designed based on the idea of local feature integration: local features in the input are integrated gradually and classified in the higher layers. In particular, the Neocognitron proposed by Fukushima in 1980 [19] is a hierarchical cascade model used for handwritten character recognition and others pattern recognition tasks that was the precursor of modern convolutional neural networks. In the following years, the idea of local feature integration was further developed and integrated into more complex structures. First with LeNet [43] and later with AlexNet [40], the successes and the enthusiasm of the scientific community have become more and more widespread and today convolutional neural networks are everywhere.

Traditional applications in the research field such as classification and retrieval have subsequently expanded into more advanced applications such as detection and segmentation and then flow into enterprise applications such as the famous self-driving cars [39]. Applications of this type have been so successful that they involve companies like NVIDIA for the development of dedicated hardware called GPUs in order to optimize computing performance (Section 2.4). Currently the most innovative development scenarios involve applications related to image captioning, pose estimation, generative models and neural style transfer [55, 39]. Researchers work closely with the most innovative universities and companies around the world to create valuable projects with cutting-edge technical solutions.

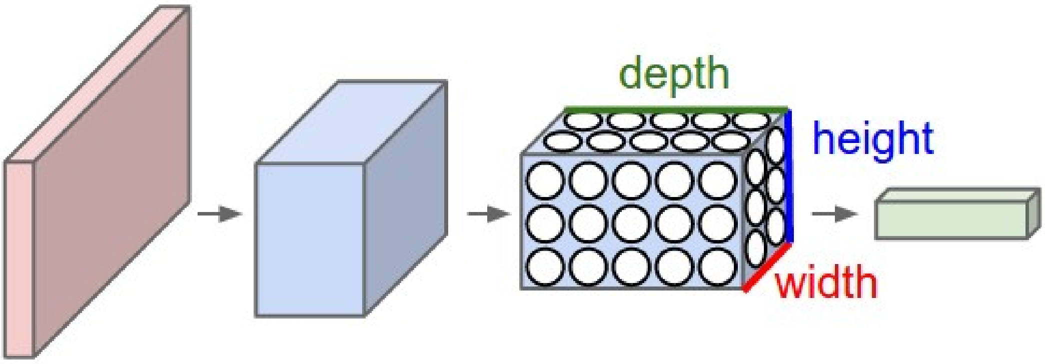


Figure 2.10: Convolutional neural network structure [39].

2.3.1 Traditional CNN Architectures

Convolutional networks are neural networks that use convolution in place of general matrix multiplication in at least one layer. This type of operator is traditionally used in fields such as computer vision and more generally signal processing as a form of filtering. Assume an input $I \in \mathbb{R}^2$ and a function $K \in \mathbb{R}^2$ called kernel: the discrete convolution S of I and K is

$$S(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n).$$

Other types of layers in addition to the convolutional ones participate in the architectural stack of modern CNNs. In particular, convolutional neural networks are made up of convolutional, pooling and fully connected layers [39]. Other layers (e.g., dropout, normalization, flatten) are widely used both to solve a specific case-study and to optimize the whole architecture. Below we will analyze the main layers and the current trends.

Convolutional (CONV) Layer

From the logical point of view, a convolutional layer [39]

- accepts an input volume of size $W_1 \times H_1 \times D_1$;
- specifies four hyperparameters: the number of filter K, their spatial extension F, their stride S and the amount of zero padding P;
- produces an output volume of size $W_2 \times H_2 \times D_2$ where $W_2 = (W_1 - F + 2P)/S + 1$, $H_2 = (H_1 - F + 2P)/S + 1$ and $D_2 = K$;
- introduces $F \times F \times D_1 \times K$ weights and K biases as learning parameters.

The convolutional layer's parameters consist of a set of learnable filters. Each filter in the forward pass is slided across the input volume computing the dot products between the input and the filter's entries. As a result, in the forward pass, for every filter an activation map that gives the response of that filter at every spatial input location is produced. Therefore, the activation maps are stacked together along the depth dimension to produce the output volume – Figures 2.11 and 2.12. Through this type of architecture, scientific studies reveal that the network will learn kernels that activate when they encounter specific visual patterns (like a change of color or orientation) in the lower layers to get to distinguish semantically stronger features in the higher layers.

The convolution operation and the concept of hierarchical organization of the structure of convolutional neural networks, over just fully connected layers, lead to particular useful properties.

Let's briefly illustrate these important properties with an example [39].

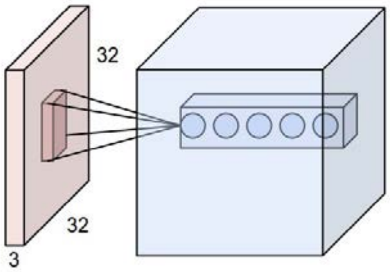


Figure 2.11: Convolutional layer [39].

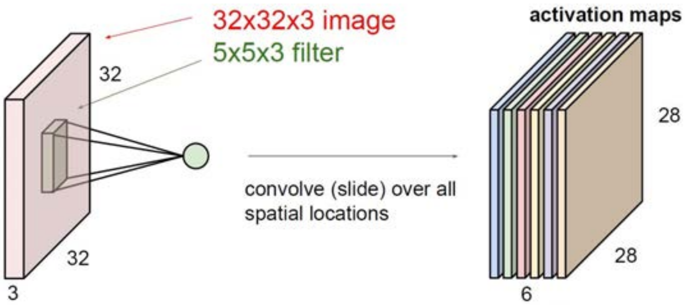


Figure 2.12: Convolutional layer example [39].

Example 1:

Consider a small RGB image $I \in \mathbb{R}^{32\times32\times3}$. Let's imagine now, regarding the concept of hierarchical representations of features, to apply a function that allows to reduce the size of the input in order to obtain more hierarchically high-level features and call this output $O \in \mathbb{R}^{28\times28\times6}$. If, therefore, to construct this architecture, we used a classic feedforward neural network, we would obtain $32 \times 32 \times 3 \times 28 \times 28 \times 6 \cong 14M$ learning parameters. Today it is not impractical to train neural networks with even more than $14M$ parameters, but consider that this is just a pretty small image and a small architecture. In this context, practical cases with images $I \in \mathbb{R}^{1024\times768\times3}$ would be unmanageable with such a large amount of parameters to learn. However, if we look at the learning parameters of the convolutional layer that could map $I \in \mathbb{R}^{32\times32\times3}$ in $O \in \mathbb{R}^{28\times28\times6}$ it is sufficient to use 6 kernels $K \in \mathbb{R}^{5\times5}$ which is equivalent to just $5 \times 5 \times 6 \cong 150$ learning parameters.

In particular, the main reasons that led to the success of convolutional neural networks are as follows [39].

Parameter sharing. The same parameter is used for more than one function in a model. This is motivated by the fact that a feature detector (e.g. a convolutional kernel) that is useful in one part of the image is probably useful in another part of the image.

Sparse interactions. In each layer, an output value depends only on a small number of input values.

The combination of the previous properties has the consequence that a much lower number of learning parameters is required and the convolutional neural network architecture better captures a translation invariance property. The property stems from the fact that an image shifted a few pixels should result in pretty similar features.

Pooling (POOL) Layer

Another important layer in moderns convolutional neural networks is the Pooling layer. From a mathematical point of view, this layer [39]:

- accepts an input volume of size $W_1 \times H_1 \times D_1$;
- specifies two hyperparameters: the spatial extension F and the stride S;
- produces an output volume of size $W_2 \times H_2 \times D_2$ where $W_2 = (W_1 - F)/S + 1$, $H_2 = (H_1 - F)/S + 1$ and $D_2 = D_1$.

This operates as a form of downsampling over each activation map computed by the convolutional layers to make the representations smaller and more manageable. Traditional downsampling operations for pooling layer consist in *max*, *avg* and *L2-norm*. Among these the most used is the *max* operation since it performs better in practical cases [39]. However, although this layer has

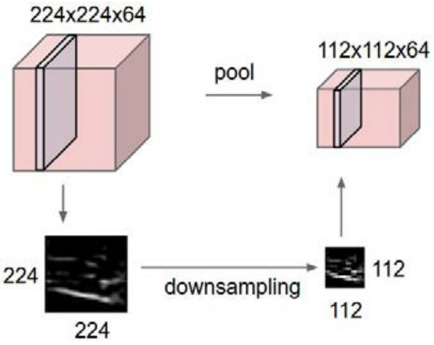
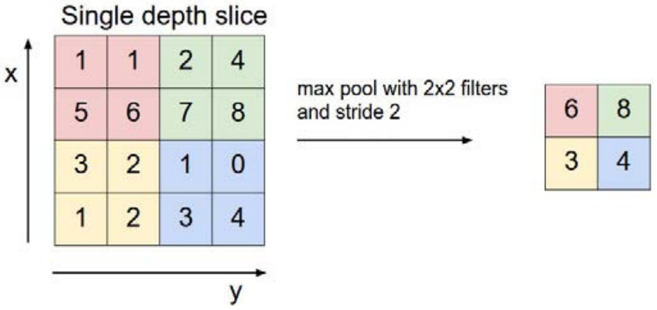


Figure 2.13: Pooling layer [39].



1	1	2	4
5	6	7	8
3	2	1	0
1	2	3	4

6	8
3	4

Figure 2.14: Max Pooling layer example [39].

some nice properties as it achieves almost scale-invariant features representation and reduces the number of learning parameters, the trend is to get rid of it [66]. In fact, it is possible to reduce the size of the representations using just convolutional layers with larger stride, thus avoiding the information loss due to the downsampling operation.

Another trend that has a different purpose but arises from similar observations is to eliminate, in addition to the pooling layers, even the fully connected ones [39]. The only difference between fully connected and convolutional layers is the fact that in the latter case neurons are connected only to a local input region and many of them share parameters. However, the function of both kinds of layers is identical since they both compute dot products, hence it is possible to convert fully connected layers into convolutional ones. Consequently, a fully convolutional [48] architecture would be obtained that allows to obtain higher computing performance.

Recently, in addition to the previous trends we see many other design choices that undermine the conventional paradigms in favor of more intricate and different connectivity structures such as the inception module of GoogLeNet [67] and the residual connections in modern ResNets [29]. However, although there are many topologies of convolutional neural networks and the possibilities of choice are very wide, in most practical scenarios it is rare to train and build a completely customized architecture. In fact, in most cases it is sufficient to fine-tune an existing pre-trained network with respect to the problem of interest [39, 78] thus reducing training and development times and achieving satisfactory performance.

2.3.2 The CNN Evolution

The structure of modern convolutional neural networks has evolved from the architecture proposed in 1988 by LeCun[43], who exploited the traditional paradigm described above. In this evolution there have been a series of experimental studies that have refined the structure optimizing the performance in terms of precision of the inferences and number of parameters, succeeding to beat the human eye [63] in 2015.

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is an international challenge that is run annually since 2010 and aims to measure the progress of computer vision algorithms for image classification and object detection at large scale [62]. This competition follows a previous smaller-scale competition, PASCAL VOC [16], established in 2005, which proposed similar objectives and was based on a dataset containing about 20 thousands images for 20 classes. ILSVRC is based on the ImageNet [40] visual database, that contains more than 14 million images for more than 20,000 classes. A trimmed list with 1000 non-overlapping classes is customarily used for the competition.

It is possible to follow in detail the succession of architectures in ILSVRC monitoring the structural trends and the improvements that each case study has provided [39]. Results are generally characterized by the analysis of the performances both of the inferences accuracy and the computational efficiency. A recent analysis [7] highlights other aspects from an practical point of view by measuring the power consumption required to train such networks. Below we will examine the main contributions to the establishment of modern convolutional neural networks.

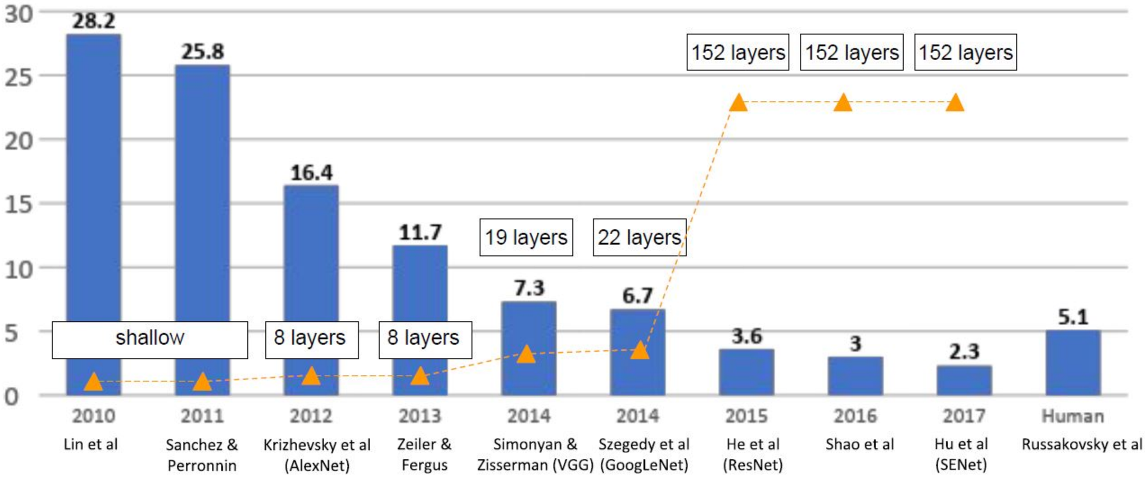


Figure 2.15: ImageNet Large Scale Visual Recognition Challenge winners [62, 39].

AlexNet

The deep learning revolution starts in 2012 with the AlexNet neural network [40]. The network presents an 8-layer structure hierarchically organized. This historically presented a whole series of hardware-side memory issues that required the neural architecture to be split into 2 distinct GPU GTX 580s. From the input image the features were extracted passing through a convolutional backbone that gradually presented semantically stronger features according to the principles described above. From the last layer of the backbone, 3 fully connected layers were connected for the classification of the 1000 classes mandated by ILSVRC'12. AlexNet was the first winner of ILSVRC to be based on a convolutional architecture with a 16.4% inference error and this was followed by many others.

VGGNet

Although the good results obtained from the improvement of AlexNet hyperparameters presented by Zeiler and Fergus in 2013 [79], even better results were obtained with deeper networks. In particular, Simonyan and Zisserman in 2014 [65] proposed a 19-layer network architecture obtaining a 7.3% error. The network was based on the concept that a stack of three 3x3 convolutional layers has the same receptive field as one 7x7 convolutional layer. The main advantage, with the same receptive field, was that with the increase in depth there was also an increase in non-linearities with consequent increases in performance. However, a downside of the deeper architecture is the increased number of parameters and consequently of memory consumption. According to this last constraint it is possible to further optimize the architecture, with no performance downgrade, by

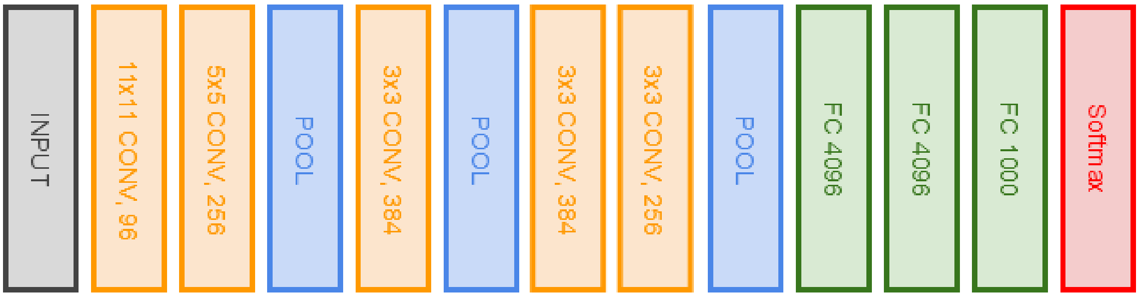


Figure 2.16: AlexNet architecture [39, 40].

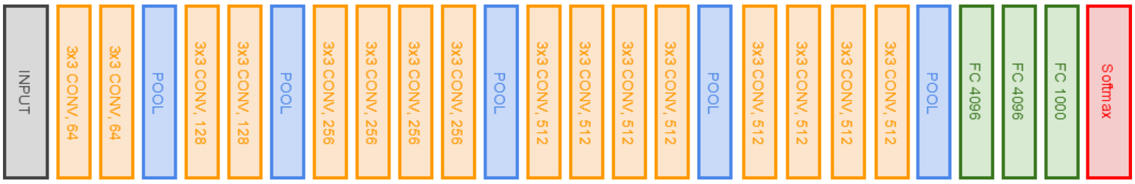


Figure 2.17: VGGNet architecture [39, 65].

removing the last fully connected layers, thus reducing the number of parameters and propelling for an almost fully convolutional architecture.

GoogLeNet

A network that demonstrates both good results from the accuracy point of view and from the computational efficiency is GoogLeNet, proposed by Szegedy et al. in 2014 [67]. In particular its main contribution has been the development of the *Inception Module* and the removal of fully connected layers that dramatically reduced the number of parameters in the network. The inception module was designed in favor of a good local network topology that apply parallel filter operations on the input from previous layers with multiple receptive field sizes for convolutions (1x1, 3x3, 5x5) and a 3x3 max pooling operation concatenating together all filter outputs depth-wise. Since this is very inefficient computationally, 1x1 convolutional layers are applied as a form of bottleneck to reduce feature depth. The full architecture is a 22-layer structure that stacks inception modules on top of each other from a first convolutional stem network to the softmax classifier output. Auxiliary classification outputs are used to inject additional gradient at lower layers.

GoogLeNet has been the ILSVRC'14 classification winner with a 6,7% error. Newer solutions as Inception-v4 [68] optimize the original version, further increasing performance.

ResNet

Since 2015 there have been a series of networks based on even deeper architectures than the previous ones, and achieving even better results. The contributions of He in 2015 [29] led to the establishment of a 152-layer network (ResNet) that outperforms previous models with a 3.6% error. The network, which takes its name from the technique used, features the famous residual connections technique to optimize the training process. In particular, it has been observed that

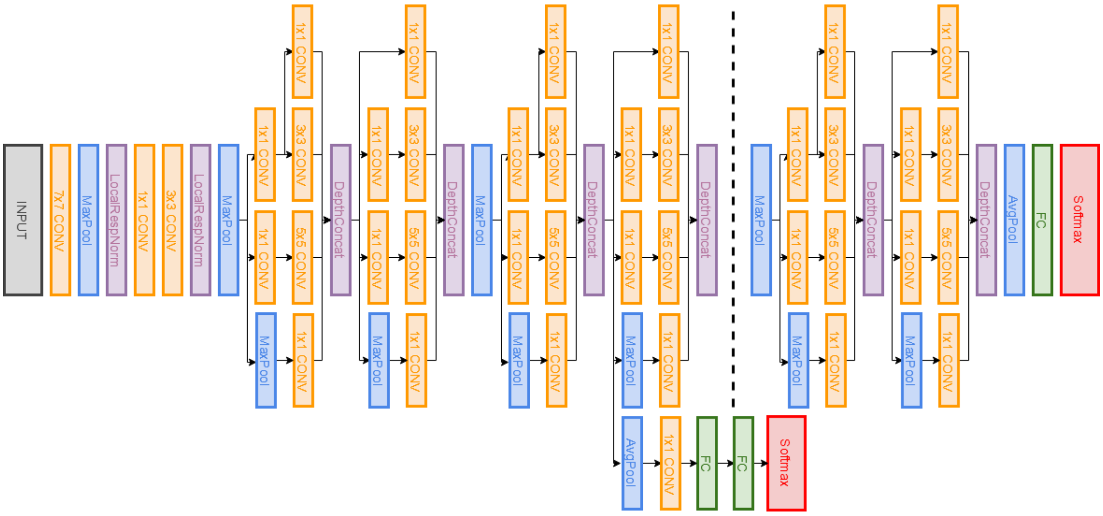


Figure 2.18: GoogLeNet architecture [39, 67].

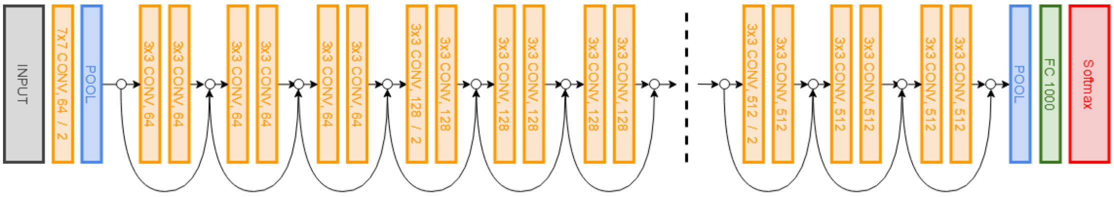


Figure 2.19: ResNet architecture [29, 39].

stacking deeper convolutional layers on a plain convolutional neural network exhibits worse performance both on training and test error with respect to a simpler model. This phenomenon, which may seem to be due to overfitting, is instead due to the fact that deeper models are harder to optimize. In fact, given the insight that deeper models should be able to perform at least as well as simpler model, the solution to the optimization problem has been to copy the learned layers from the simpler model and to set additional layers to identity mapping. The technique fits a residual mapping to upper layers instead of directly trying to fit a desired underlying mapping.

The full ResNet architecture stacks together residual blocks composed by two 3x3 convolutional layers from a first convolutional layer at the beginning to the softmax output. Bottlenecks 1x1 convolutional layers further improve efficiency in a GoogLeNet-like fashion.

Residual networks are currently the state of the art of convolutional neural networks for image classification, and are a good default choice for practical scenarios. The structures of more recent convolutional neural networks are inspired by this model and develop innovative features with new ResNet-based topologies [77]. A significant part of the world-wide research in this area focus on the design of residual connections to improve the gradient flow and investigate the effects of wider and deeper models on output results [39]. Even newer trends concern the design of meta-learning algorithms to further improve performances.

2.3.3 Transfer Learning

Transfer learning is a technique that aims to transfer knowledge from an already learned task to a new one. This technique has been studied and quantified for modern convolutional neural networks in [78]: this work investigated artificial neurons transition from general to specific in deep convolutional neural networks. In particular, in the AlexNet case it has been observed that, while first-layer features are general purpose, last-layer ones are task-specific. The technique has been used by subsequent works [23] that confirmed the results, and is currently very popular in the deep learning field. Nowadays, in fact, there is a tendency to avoid training convolutional neural networks from scratch. This is because it is very difficult to have large datasets. In practice [39], convolutional neural networks are pre-trained on large generic datasets and subsequently either fine-tuned or used as feature extractors for the task of interest.

Feature extraction. The technique aims to remove the last fully connected layer of a convolutional neural network pre-trained on a large dataset (e.g. ImageNet) to treat the rest of the network as a feature extractor for a new dataset. Subsequently, a task-specific classifier (often, a linear one) is trained on the features extracted from the network.

Fine-tuning With fine-tuning, the on-top classifier of the pre-trained convolutional neural network is not only replaced and retrained, but some of the previous layers are retrained too. This is due to the fact that, as described in [78], convolutional layers become progressively more specific to the trained dataset.

Moreover, as we will discuss later in Section 2.4, a zoo of pre-trained models is available for developers to take advantage of open-source contributions.

The applicability of the Transfer Learning technique depends several factors, and chiefly the size of the new dataset and its similarity with the original one [39]. The size of the dataset implies the possibility or not to fine-tune due to overfitting concerns. Depending on the size of the dataset, it

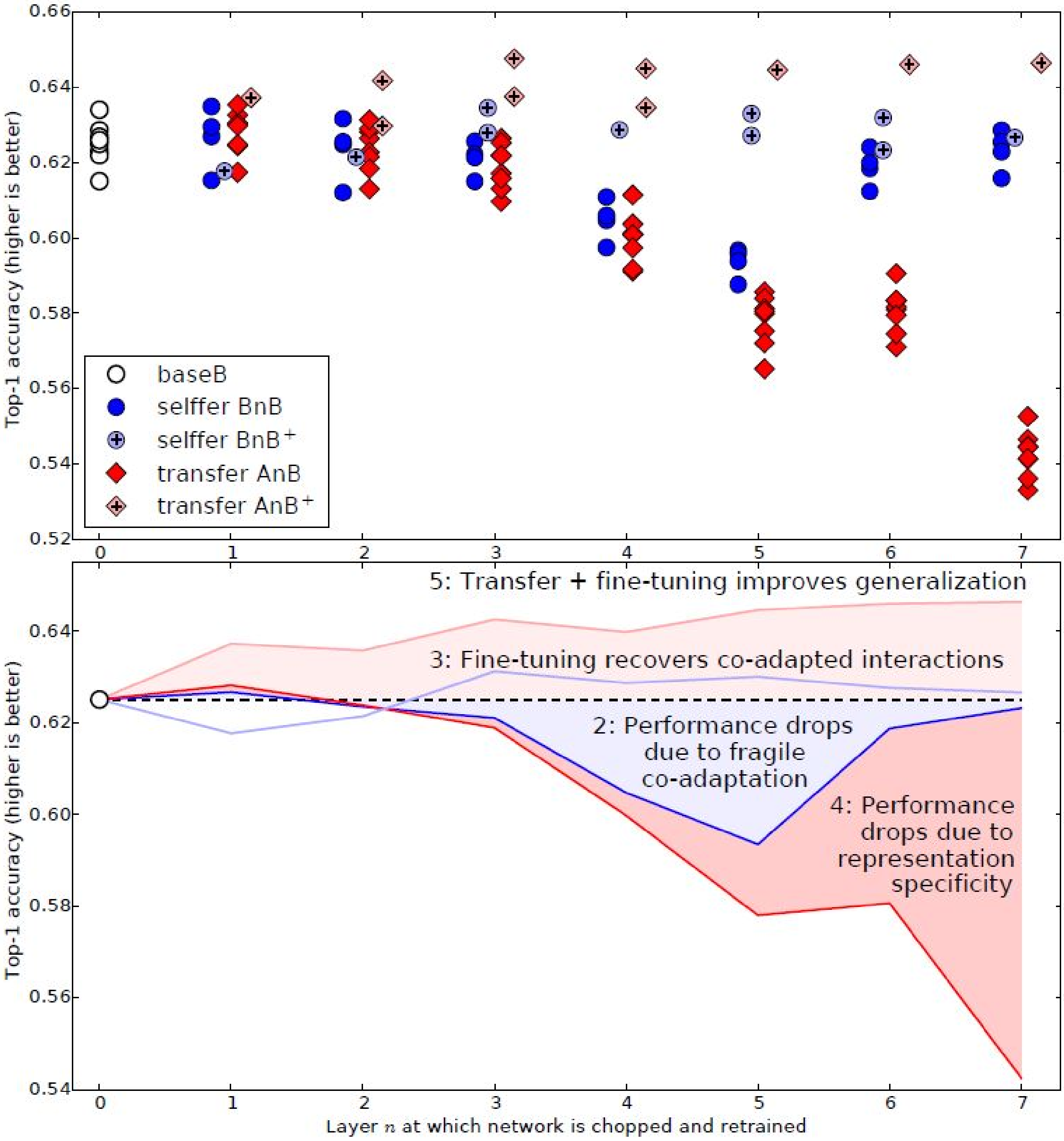


Figure 2.20: Generality over specificity measured as transfer performance on AlexNet [78]. For the test two tasks, A and B, are created splitting the ImageNet classes into two groups each containing approximately half of the data. The network trained on these base tasks are called *baseA* and *baseB*. Starting from these networks, using transfer learning, new networks have been trained using a progressively more extensive fine-tuning. In particular, a selffer *BnB* network is a network where the first n layers are copied from the *baseB* network and frozen while the subsequent layers are retrained on the B dataset. A transfer *AnB* network is a network where the first n layers are copied from the *baseA* network and frozen while the subsequent layers are retrained on the B dataset. For the *BnB*+ and *AnB*+ networks above considerations apply, but all the layers learn.

is indeed possible to retrain a larger portion of the network in order to achieve more task-specific results. The similarity of the dataset implies the need to perform a more or less extended finetune. In fact, if the reference dataset were very different it would not be possible to obtain a good generalization by using the neural network as a simple feature extractor.

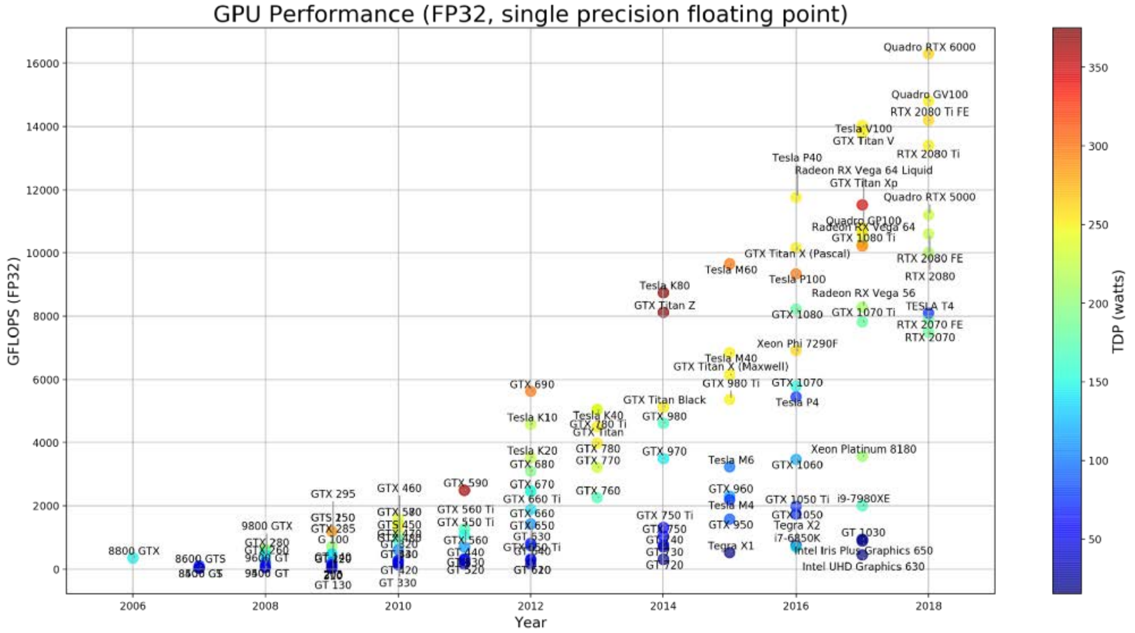


Figure 2.21: Hardware performance FP32 over years [39].

2.4 Hardware, Frameworks and Datasets

Modern neural networks are the result of a continuous technological improvement process made possible by an evolution of dedicated hardware and software. Deep learning solutions can be found today in our computers, smartphones, tablets and servers of the most innovative companies around the world. In particular, while hardware progress sees the development of more and more specialized architectures through the now famous NVIDIA GPUs, software development is entrusted to frameworks such as TensorFlow [51] and Pytorch. There are also many contributions from the open source community that make new dataset and pre-trained models available every year that can be used by anyone for the development of customized solutions in a short time.

Below, we will briefly present the history of technological improvements for hardware, software and datasets.

2.4.1 Hardware: from CPU to GPU/TPU

There are two distinct phases that make it possible to use modern neural networks: Training and Inference, where the network is tuned to solve a specific task and where the network is actually used as an inferential engine, respectively. The training phase, in particular, is much more computationally demanding than the inference phase, and can take days. Hardware optimization tries to improve the performance of both phases, obtaining both real-time inferences and faster training. However, while training requires almost exclusively the use of very powerful servers, the inference can easily be performed even in multi-core CPUs of modern smartphones or tablets [39].

The Central Processing Units (CPUs) are the traditional computational units that centrally coordinate all the processing tasks in computer hardware architectures. In particular, while most desktop devices mount multi-core CPUs with 2 to 8 cores, the current trend sees a further increase in the number of cores if we think of recent Intel Core i9 and AMD Ryzen Threadripper architectures. In computer servers, Intel Xeon and AMD EPYC already offer better performance with a number of cores traditionally superior to the desktop market and other useful technology such as ECC memory and multi-processor configurations. However, currently, CPUs are not the dominant technology in the hardware market of deep learning. Some approaches try to use cluster of CPUs like Intel's

Product	Cores	Clock Speed	Memory	Price	Speed
Intel Core i7-7700k	8 (hyper)	4.2 GHz	System RAM	$339	~540 GFLOP/s (FP32)
NVIDIA GTX 1080 Ti	3584	1.6 GHz	11 GB GDDR5X	$699	~11.4 TFLOP/s (FP32)
NVIDIA TITAN V	5120 CUDA, 640 Tensor	1.5 GHz	12 GB HBM2	$2999	~14 TFLOP/s (FP32), ~112 TFLOP/s (FP16)
Google Cloud TPU v2	-	-	64 GB HBM	$6.5 (hour)	~180 TFLOP/s

Table 2.1: Deep learning hardware comparison [39].

BigDL or software libraries specialized for deep learning on CPUs, but the results are not very promising and have practical use only in case there is already a CPU-based supercomputing center.

The hardware market for deep learning is currently dominated by Graphics Processing Units (GPUs). Originally created for the computer graphics industry, GPUs are specialized computational units with a number of cores two orders of magnitude higher than classical CPUs [39]. The high number of cores and the high degree of parallelism that results makes GPUs a central technology for the efficient execution of algorithms based on matrix operations and convolutions, as it is the case in deep neural networks. NVIDIA GPUs are undoubtedly the most used processing units in deep learning frameworks, while AMD GPUs don't enjoy the same popularity.

Modern deep learning systems currently see an integration of CPUs and GPUs where the CPU oversees the loading and management of the computation and the GPU is responsible for the computational load [39]. This trend is so clear that we are witnessing three distinct phenomena:

- the design of distributed computing clusters that can be used in the cloud, such as AWS, Ms Azure and GCP, and powerful supercomputers such as IBM Summit;
- the development of dedicated computing accelerators for deep learing, such as Google's Tensor Processing Unit (TPU) and Neural Processing Unit (NPU), to get faster training on servers and low latency inferences on smartphones;
- an increase in accessibility towards the creation of scalable computing infrastructures built specifically for businesses or individuals by reducing the costs of the various components.

In parallel to the development of the previous points, research also explores alternative solutions [39] based mainly on concepts such as distributed model training, neuromorphic computing and quantum computing.

2.4.2 Frameworks and Dataset Zoos

In order to make the best use of specialized deep learning hardware, it is necessary to exploit strongly engineered, parallel programming concepts that have a whole series of implications on the software side [35]. In particular, while low-level computing processes can be managed directly on GPUs using parallel computing platforms such as CUDA and optimized libraries for deep learning such as cuDNN and NCCL, high-level modern frameworks offer a fundamental abstraction that allows the developer to focus directly on the design of the network architecture.

Almost every year a new framework is released for generic or task-specific deep learning [35]. This is due to the fact that the major global entities involved in innovative projects in AI benefit from specialized software. TensorFlow, PyTorch, Keras, Caffe, MXNet, CNTK are just some of the names that are mentioned in the development portfolios of modern deep learning applications [35, 39]. While some frameworks like Caffe, Theano and Torch were born in the academia, the most famous ones such as TensorFlow, PyTorch and CNTK – that in some cases derive from the previous ones – were developed by large specialized software companies such as Google, Facebook and Microsoft.

TensorFlow, PyTorch and other deep learning frameworks, including Microsoft Cognitive Toolkit, are released under open-source licenses [39]. More generally, the deep learning world follows a predominantly open-source philosophy and many works by large companies such as Google, Facebook

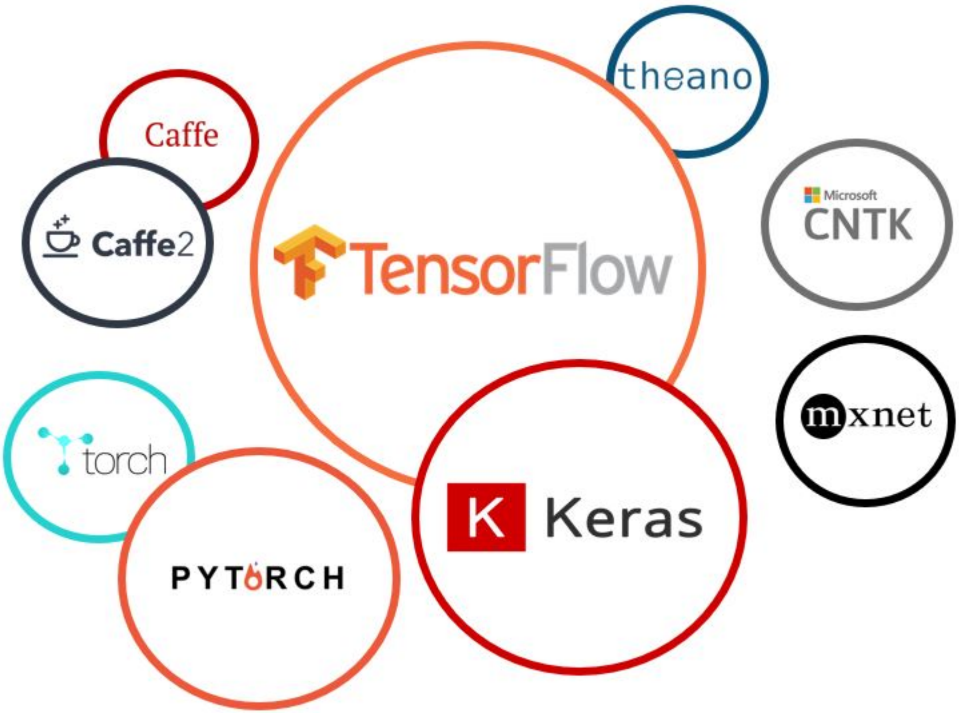


Figure 2.22: Main machine learning frameworks.

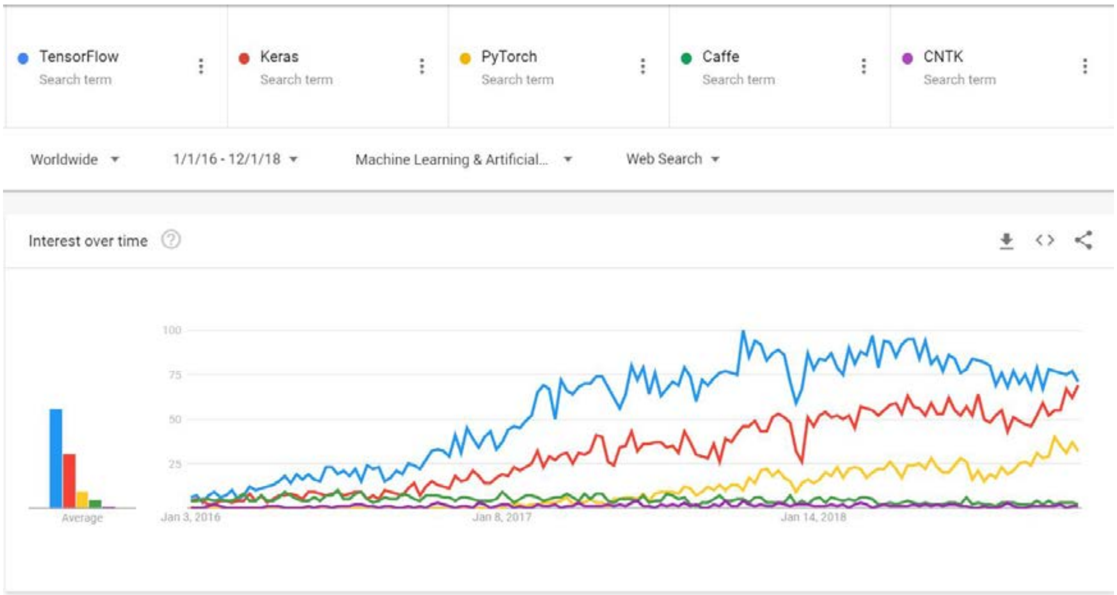
and Microsoft are published through research papers. This makes it possible to know the algorithms in detail and to exploit this knowledge on the research side proposing new innovative solutions, or on the business side with solutions targeted to customers. Research in the field moves at a fast pace and every year innovative discoveries are proposed that promptly provide new interpretations and contribute to everyday applications.

In particular, beyond the specific software libraries, frameworks generally provide:

- portfolios of high-level APIs that extend basic features;
- pre-trained models and training configurations;
- datasets to train the models from scratch.

The high-level APIs are traditionally linked to a specific domain of interest such as computer vision or natural language processing, and allow a very fast prototyping of the neural networks. These can be developed directly by the framework's developers (eg. Google Brain) such as TensorFlow's object detection APIs [32] or released open-source by third parties as for example the Matterport's [1] implementation of the Mask R-CNN algorithm [30] in TensorFlow. Pre-trained models, on the other hand, are grouped in zoos and can be used directly for inferences or can be exploited to do transfer learning or network surgery to solve a specific task. In this, the datasets play a key role in models training and in the world of deep learning. The latter assume both a development role in case the specific application requires to perform fine-tuning or domain transfer, and a role of verification of models performances as a form of certification.

Each framework has distinct characteristics that make it usable in a particular domain of interest. It is therefore possible to compare the various architectures regarding hardware support, software compatibility, computational efficiency, ease of use, documentation and interest from the community [64, 4]. In particular, although all points are touched differently from each framework, what currently emerges from a cross analysis is the popularity of TensorFlow, Keras, and PyTorch. The interest of the deep learning community is diversified and rewards the TensorFlow documentation and guidelines, which makes it an excellent tool in production, and the architectural style



Worldwide ▾ 1/1/16 - 12/1/18 ▾ Machine Learning & Artificial... ▾ Web Search ▾

Figure 2.23: Main machine learning frameworks trends [25].

of PyTorch, which makes it a great research tool. Keras, instead, is a Python library that can be used on top of TensorFlow and other frameworks (PyTorch is not among them, however), and allows an agile and lightweight prototyping of models thanks to its high-level APIs.

In what follows we will focus more on TensorFlow for its ability to support deep learning models at scale, but we will make comparisons with PyTorch to highlight its peculiarities and software structure.

TensorFlow

TensorFlow is a software library for machine learning developed by Google Brain and released under the Apache 2.0 open-source license [51]. The library provides tested and optimized modules useful in the implementation of AI algorithms for different tasks and is widely used in many commercial products of well-known international companies. TensorFlow provides both low-level and high-level APIs [51], the latest available via Keras. It is used to build deep learning models to make inferences on structured, unstructured and binary data such as text, images and audio, as well as to build generative models – through the famous GANs, for instance. It is possible to design both research applications with Eager Execution and large-scale deployments in production environments through Estimators.

TensorFlow [51] supports the major 64-bit operating systems Windows, Linux and MacOS and provides native APIs in Python, C/C++, Java, Go and third-party APIs available in C#, Julia, Ruby and Scala. There is also an ongoing project to integrate TensorFlow directly in Swift. Different projects are also dedicated to web and mobile development, respectively TensorFlow.js, which allows training and deploying of models in web applications through Node.js, and TensorFlow Lite, which represents the official solution for running models on mobile and embedded devices such as Android and iOS.

From a technical point of view, the key principles on which TensorFlow is based are [35]:

- the definition of computational graphs composed of n-dimensional base units called tensors that abstract the underlying architecture and can be run on CPUs or GPUs;

- the automatic computation of the gradient and other mathematical utilities for the design of computational graphs.

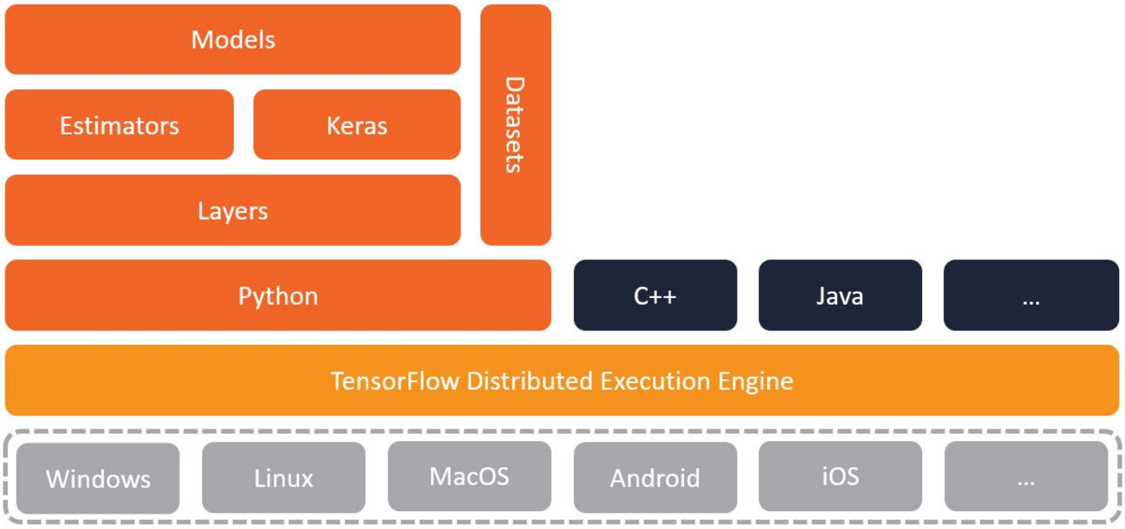


Figure 2.24: TensorFlow architecture [51].

For computational graphs, TensorFlow traditionally follows the "define and run" paradigm that allows to define a static computational graph and then use it in every iteration. The graph can be defined node by node through the low-level APIs [35], or sequentially as a succession of predefined layers through the high-level routines included in Keras. This is opposed to PyTorch's vision where computational graphs are dynamically defined according to the "define by run" paradigm [35]. If the definition of static computational graphs allows the underlying framework to perform a series of optimizations before execution, the dynamic variant allows greater flexibility for algorithms that are dynamic by definition such as modern recurrent neural networks. However, this traditionalist vision was overcome with TensorFlow 1.7 through the Eager Execution that allows the definition of dynamic graphs overcoming the previous problems.

In addition to the base library, a suite of specific and general purpose software development tools and APIs are available for developers [51]: Tensorboard for learning visualization, monitoring and model inspection; TensorFlow Debugger for a specialized debugging; TensorFlow Serving for the deployment of deep learning models in production environments through the classic client-server architecture; and other utilities. It is also possible to take advantage of the multi-GPU distributed training features and of more advanced utilities for training on Google Cloud TPUs [39, 51].

2.4.3 International Challenges and Datasets

Below we will give a brief overview of the most famous datasets and international challenges that see them involved. We will only analyze datasets in the object detection field, as this deep learning project falls into such field. Dedicated datasets in the retail systems analytics field will be treated later in the chapters that describe the system model.

Every year new datasets over specific contexts are created; new international challenges arise and researchers from all over the world challenge each other to design cutting-edge targeted solutions. The major international challenges in computer vision are mainly focused on topics such as image classification, object detection and semantic segmentation. Parallel to the main tasks, new challenges arise that integrate the same tasks in different contexts. This is the case for instance segmentation, image captioning and many other challenges.

Currently, the main datasets that see the greatest number of international challenges derive from a succession of datasets in the computer vision field started in 1998 with the famous MNIST dataset for handwritten digits recognition [42]. After MNIST, other datasets, such as ImageNet [40] and PASCAL VOC [16], respectively in the image classification and object detection fields, have been created. These have subsequently integrated other types of challenges and ImageNet

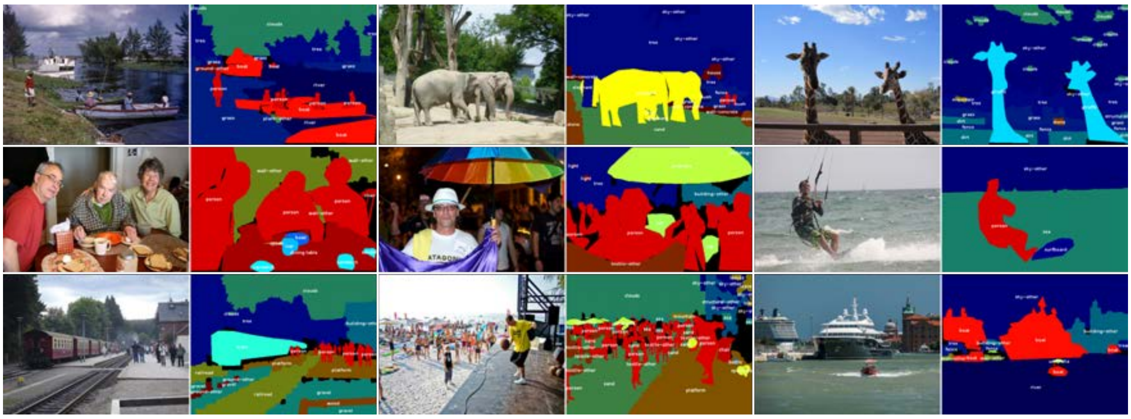


Figure 2.25: Images and annotations from Microsoft COCO.

Name	Description	Size	Class	Tasks	Format
MNIST	Handwritten digits recognition	60K	10	Image classification	28 × 28 Gray
Fashion MNIST	Fashion products recognition	60K	10	Image classification	28 × 28 Gray
CIFAR-10	Small images, various classes	60K	10	Image classification	32 × 32 RGB
CIFAR-100	Small images, various classes	60K	100	Image classification	32 × 32 RGB
Caltech-101	Small images, various classes	9K	101	Image classification	Small RGB
Caltech-256	Small images, various classes	30K	256	Image classification	Small RGB
PASCAL VOC	Various themes	11.5K	20	Image classification, object detection, instance segmentation, action classification	Small RGB
ImageNet	Visual Recognition at scale	14M	21K	Image classification, object detection	RGB
Ms COCO	Complex everyday scenes of common objects in natural context	330K	171	Object detection, semantic segmentation, instance segmentation, keypoint detection	RGB
Open Images	Unified image classification, object detection, and visual relationship detection at scale	9M	19.8K	Image classification, object detection, visual relationship detection	RGB
SUN	Very large scene and object recognition database	131K	4.7K	Object detection, instance segmentation	RGB
Cityscapes	Semantic understanding of urban street scenes	25K	30	Object detection, instance segmentation	RGB
DAVIS	Densely annotated video segmentation	10.5K	-	Object detection, instance segmentation, video segmentation	RGB
LabelMe	Dynamic dataset of annotated natural scenes	187K	-	Object detection, instance segmentation	RGB
Visual Genome	Connecting language and vision with dense image annotations	108K	33.9K	Object detection, image captioning	RGB

Table 2.2: Famous deep learning image classification and object detection datasets [11].

currently in addition to image classification is used for object detection while PASCAL VOC has also been able to integrate instance segmentation. Starting from these datasets, in 2014 Microsoft COCO – Common Objects in COntext – has been created [46]. This has resized the concepts of instance segmentation started by PASCAL VOC at scale and is currently the main reference dataset together with ImageNet all over the world.

Chapter 3

Object Detection: State of the Art

With the recent advancements in deep learning, modern object detection methods have become very popular for the outstanding results achieved. These are applied in different contexts in the industrial field as the core of the most innovative projects by the main market players. Video surveillance, self-driving cars, robotic process automation and retail systems analytics are just some of the recent declinations of these algorithms.

This chapter contains a technical overview of all the main themes in the object detection landscape. The most performing object detection algorithms as well as activities ranging from classification to semantic segmentation that often act as an integration to the proposed algorithms will be treated. The contents of this chapter will be the core innovation engine of subsequent developments in the retail products analytics area proposed by our work.

3.1 An Object Detection Review

Object detection [39] is currently one of the main tasks of computer vision. This is the natural evolution of image classification, another very well known task in the computer vision field that provides for the assignment of a specific label to an image to characterize its content. Object detection differs from the latter because it aims to determine, for an arbitrary image, both the label and the bounding box that defines each entity in the scene. Besides object detection, other tasks have assumed great importance nowadays. These are semantic segmentation and instance segmentation which provide to make per-pixel predictions to identify for a generic image respectively all the pixels belonging to a given class and all the pixels belonging to a given class instance. Other declinations see the extension of these tasks from the 2D world to the 3D world, introducing also advanced domain-specific challenges as video streaming understanding.

Given the semantic gap between what a human being is able to understand and what a computer is able to process, a series of issues are defined that make these tasks extremely challenging. In particular, computer vision algorithms traditionally have troubles dealing with [39] viewpoint variation, different illumination and deformation conditions, occlusion, intraclass variation and many other challenges. Therefore actually there is no obvious way to hard-code algorithms to solve the previous tasks except in a restricted domain.

Object detection – and, more generally, computer vision algorithms – have evolved over time from heuristic analytics techniques based on empirically defined rules to reach the modern data-driven approach [39]. The latter aims to train a classifier on a set of reference images and annotations called dataset and currently is very successful in terms of results achieved. The following constitutes a basic yet standard-compliant overview of the evolution of object detection approaches.

Geometric invariants. This class of algorithms exploits invariant geometric characteristics of classes of distinct geometric objects such as circles and rectangles to identify specific patterns in the images. Hough Transform and Generalized Hough Transform algorithms are part of this algorithmic class [36]. The latter in particular constitute a template matching approach

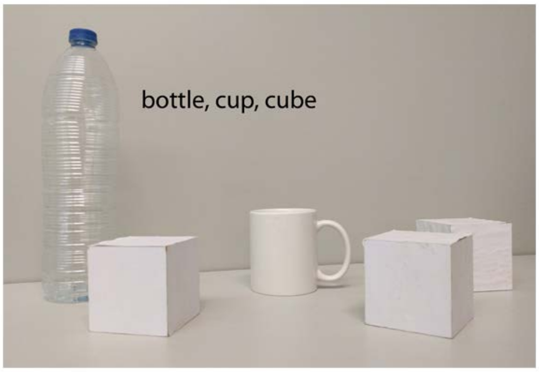


Figure 3.1: Image classification [20].

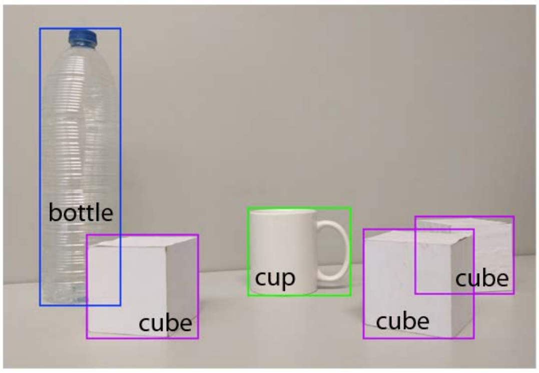


Figure 3.2: Object detection [20].

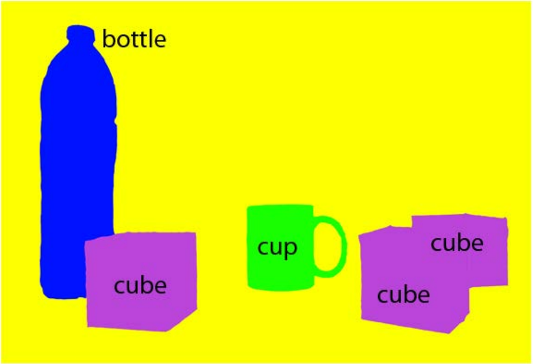


Figure 3.3: Semantic segmentation [20].

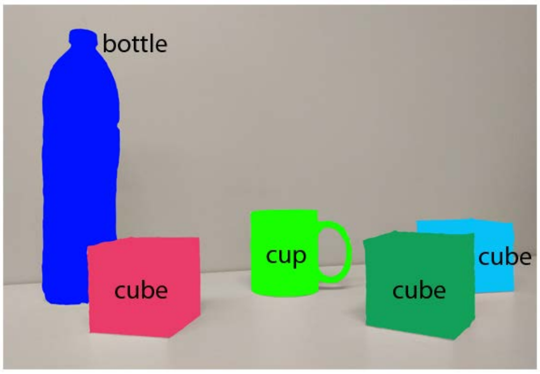


Figure 3.4: Instance segmentation [20].

to determine which objects belong to the same class through an evaluation procedure of each candidate in a parametric space.

Feature based. Starting from an image it is possible to identify special points called keypoints through features detectors, as well as to associate these points based on the surrounding area through features descriptors. The detector provides a stable keypoints identification, insensitive to light changes and geometrically accurate. The descriptor, based on information from color, texture and orientation, instead provides a vector representation of a specific part of an image from the keypoints identified by the detector. The main application of the previous algorithms consists in comparing specific patterns in different images through a matcher. SIFT [49], SURF [5], HOG [14]: these names are all part of a famous pool of algorithms for object detection based on features.

Bag of features. The bag of features approach [41] is inspired by the famous bag of words method and combines both the data-driven approach and the feature-based one. The initial step of this approach involves learning a visual vocabulary based on features extracted from feature-based algorithms. Subsequently, it is possible to perform the classification and the detection by appropriately aggregating the visual vocabulary identified in the image through further data elaborations.

Deep learning. The data-driven approach has been developed over the years attaining increasingly higher goals. This was initially developed as a classification problem based on the concept of proposals and subsequently it has recently been formulated in an alternative way as a regression problem [39]. The first category includes the two-stage region-based approaches that perform entities detection on the basis of candidate image portions called regions. The second solution instead belong to single-stage approaches that simplify the can-

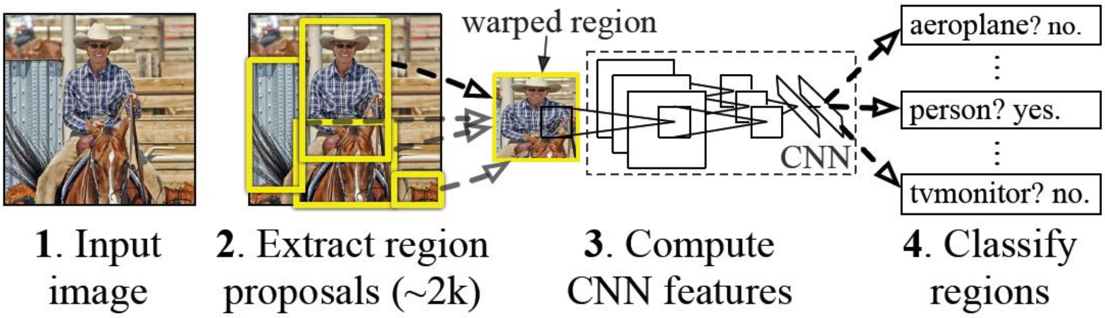


Figure 3.5: R-CNN pipeline [23].

didates proposal problem and directly execute the inferences on image areas defined a priori. Both solutions are currently used and constitute the state of the art of object detection algorithms.

The following sections contain a review of the most promising object detection algorithms adopting the deep learning approach [39]. The collection does not aim to be exhaustive but contains the main results that have led the evolution towards modern approaches. A fundamental reflection point in the study of these algorithms is the experimental evaluation [27]. This is the main tool for performance evaluation and provides useful information on the design of the most innovative approaches. Experimental evaluation will be discussed in detail in the object detection field in Section 5.1 but the following sections will still provide key results and insights for efficient algorithms design. In particular, as a metric to evaluate performance, mAP – mean Average Precision – is often used to give a measure of the number of relevant objects retrieved in the test set images by the algorithms with respect to the ground truth. This is often calculated with respect to the IoU – Intersection over Union – threshold (where if not specified it means 50%) which indicates the percentage of surface of an object to be retrieved from the algorithm so that the latter's inferences can be considered positive.

In some circumstances, the ablation experiments will be briefly discussed to provide useful key insights for understanding the various architectures. These experiments, have been originally used in the field of psychology to study the effects of removing parts of the animal's brains on their behavior [73]. Actually especially in the deep learning landscape ablation studies have been adopted to gain better understanding of the networks behaviour by systematically removing its parts [23].

3.2 Region-Based Detectors

Deep learning techniques have been recently established as a way to learn features representations directly from data. This have been particularly important in the object detection field, bringing exciting results in many application areas.

Region-based detectors in this context have emerged as a powerful method for identifying entities in images and are used as the core of many practical applications [23]. These detectors exploit a double-stage approach that allows to classify predefined entities starting from the proposal of candidate regions in the image. The approach has been further refined over time reaching ever higher standards. Actually in parallel with algorithms engineering, a further field of interest sees the addition of inference branches for complementary domains such as instance segmentation or human pose estimation. Below we will discuss in detail the main steps of this evolution.

3.2.1 R-CNN

In 2014, an innovative approach was proposed based on the intuitions deriving from recent developments in the region proposals [72] and transfer learning [78] fields. The approach was able to

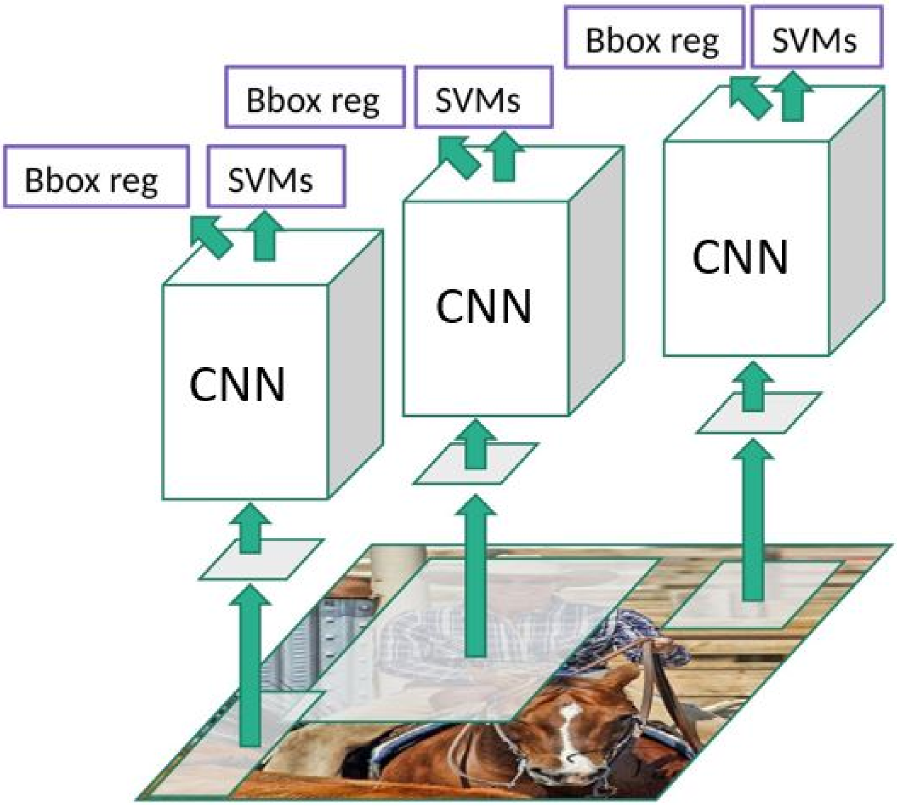


Figure 3.6: R-CNN architecture [23, 39].

improve the performances measured on the PASCAL VOC 2010 [16] dataset by 30% compared to previous algorithms, achieving a mAP of 53.7% [23].

The algorithm architecture is based on three distinct stages [23].

Region proposals. The Selective Search algorithm is responsible for proposing possible non-categoric candidate regions for a subsequent classification [72]. The algorithm exploits color and texture similarities and other image features to return blobby image regions that are likely to contain objects. Relatively fast to run, Selective Search returns about 2000 region proposals in few seconds.

Features extraction. A backbone consisting of an AlexNet [40] convolutional neural network topology is used as a feature extractor on previous warped region proposals. The use of different backbones has been further investigated in the ablation studies and positive effects have been found in other topologies.

Classification. The features extracted are subsequently classified by a collection of SVMs, one for each class in the dataset. Inferences are then aggregated through non-maxima suppression to return final predictions.

Ablation Studies:

Ablation studies by the authors also confirmed the usefulness of transfer learning [78] and further algorithmic optimizations. With regard to transfer learning, positive effects were obtained by pre-training the convolutional backbone on a large auxiliary dataset (ILSVRC2012 [62]) and subsequently adapting the CNN to the new domain (PASCAL VOC 2010 [16]) with a domain-specific fine-tuning. The results of the final classification by means of SVMs

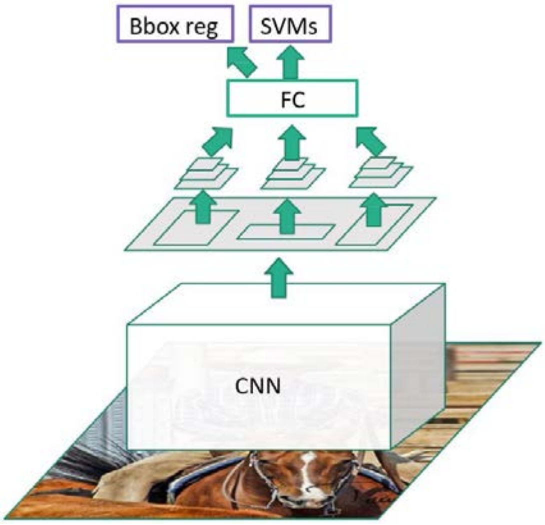


Figure 3.7: SPP-net architecture [31, 39].

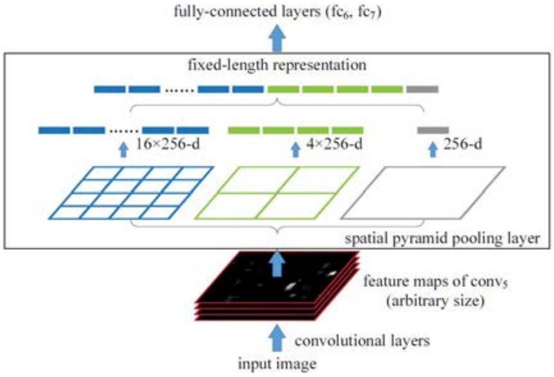

fully-connected layers (fc$_6$, fc$_7$)
fixed-length representation
16×256-d
4×256-d
256-d
spatial pyramid pooling layer
feature maps of conv$_5$ (arbitrary size)
convolutional layers
input image

Figure 3.8: SPP layer [31].

were compared with a more traditional classification by softmax and found a performance degradation in the softmax case of about 3.3% on PASCAL VOC 2007. The introduction of a further linear regression inference branch as a refinement of the bounding box coordinates previously determined by the Selective Search region-proposals algorithm has found an increase in performance on the PASCAL VOC dataset of about 3.5% compared to the base case. The threshold that regulated the degree of IoU – Intersection over Union – of the overlapping regions was a fundamental parameter to obtain good inference performances. A grid search over $\{0, 0.1, \cdots, 0.5\}$ showed good results with a 0.3 threshold.

Starting from R-CNN other algorithms have been developed that have improved various aspects of the original architecture. In particular, the inference speed has been further optimized in parallel with the possibility of performing end-to-end training, achieving higher performance standards.

3.2.2 SPP-net

SPP-net and the concept of Spatial Pyramid Pooling had been proposed by He in 2014 as an optimization of traditional convolutional neural networks [31]. The approach eliminates the constraint of fixed-size input of the convolutional neural networks generating a fixed-length representation regardless of image size and scale. In particular, the fixed-size input constraint is relative only to the fully-connected layers while it does not apply to convolutional ones. To adapt CNNs to images of any size, the Spatial Pyramid Pooling layer is introduced as an interface to the fully-connected layers that pools the output of feature maps in local spatial bins.

In particular, the usefulness of SPP-net in the object detection field has been pointed out by proposing an optimization that accelerates R-CNN from 10× to 100× at test time and by 3× at training time. Since R-CNN is slow because it does not share computations between convolutional region-proposal forward passes, the SPP method classifies each region proposal using a feature vector extracted from a shared convolutional feature map. Multiple region-proposal output feature sizes are pooled and concatenated into a fixed-size output through a Spatial Pyramid Pooling layer.

3.2.3 Fast R-CNN

Fast R-CNN [22] was proposed by Ross Girshick as a R-CNN and SPP-net optimization in 2015. According to the author, Fast R-CNN is 9× faster than R-CNN in training and 213× in inference and achieves also higher accuracy performances obtaining 66% mAP on PASCAL VOC 2012 [16].

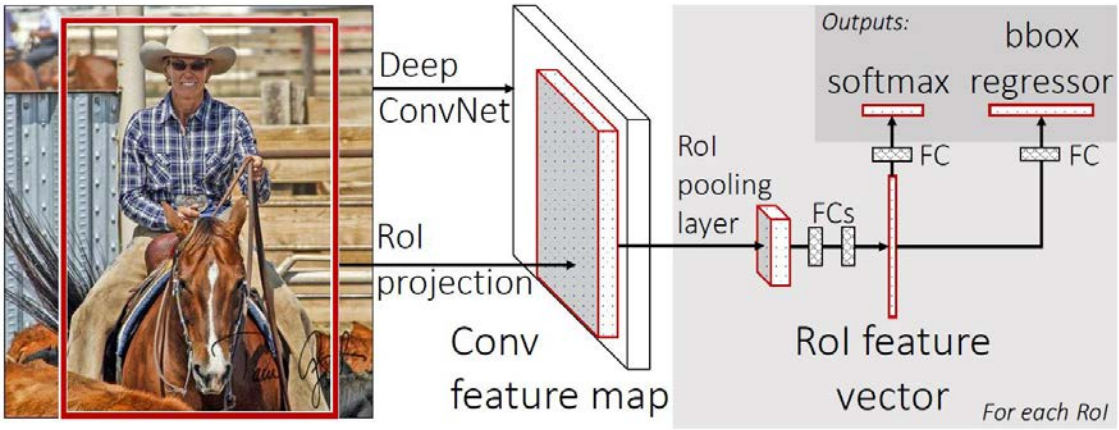


Figure 3.9: Fast R-CNN pipeline [22].

The main drawbacks of R-CNN and SPP-net underlined by Girshick concern the multi-stage training pipeline and the inefficiency in time and space recorded in the various phases of the algorithm. For the second in particular, this requires caching the features extracted from the SPP layer to fine-tune the FC layers on top and then train the SVMs and linear regressors. In fact, one of the main disadvantages of the method proposed by He is that the fine-tuning algorithm cannot update the convolutional layers that precede the SPP layer and this strongly limited the accuracy of very deep neural networks. This is due to the inefficiency introduced in the backpropagation algorithm when each ROI training sample comes from a different image, as highlighted in [22].

The improvements of Girshick [22] over SPP-net [31] are mainly training optimizations. The architecture of the network is reviewed in favor of a simpler structure that allows end-to-end and training. This is achieved using multi-task loss on softmax and bounding box regression output layers where softmax takes the place of previous SVMs. SPP layer also changes in favor of a reduced configuration called ROI Pooling layer which consists of an SPP layer with a unique pyramid level compatible with the net's first fully connected layer. Furthermore, a more intelligent fine-tuning algorithm is proposed to overcome the constraint of SPP-net relative to the update of the layers preceding the SPP one – in this case, ROI Pooling. This exploits feature sharing during training to use more than one ROI per image in each mini-batch, thus significantly reducing the training time.

Ablation Studies:

Ablation experiments are particularly interesting and highlight key properties of convolutional neural networks. The scale-invariance property has been examined in detail and experiments have preferred a ROI Pooling based architecture compared to the previous one based on Spatial Pyramid Pooling, arguing that deep convolutional neural networks directly learn scale invariance. Further studies on fine-tuning of deep convolutional neural networks (in this case VGG16 [65]) have shown that to obtain good results from the approach it is preferable to extend the range of the layers involved in training. Also other studies show that multi-task training has the potential to improve results through tasks mutual influence and that SVMs do not outperform softmax classifier learnt with fine-tuning technique. Finally, further studies have highlighted the trade-off between precision and recall taking into account the generation of a defined number of region proposals.

3.2.4 Faster R-CNN

The optimization of the region-based approach evolved further in 2015 when Ren proposed a variant of the original algorithm capable of achieving almost real-time performance [59]. The author declares a frame rate of 5 fps on GPU while achieving a mAP of 78.8% on PASCAL VOC

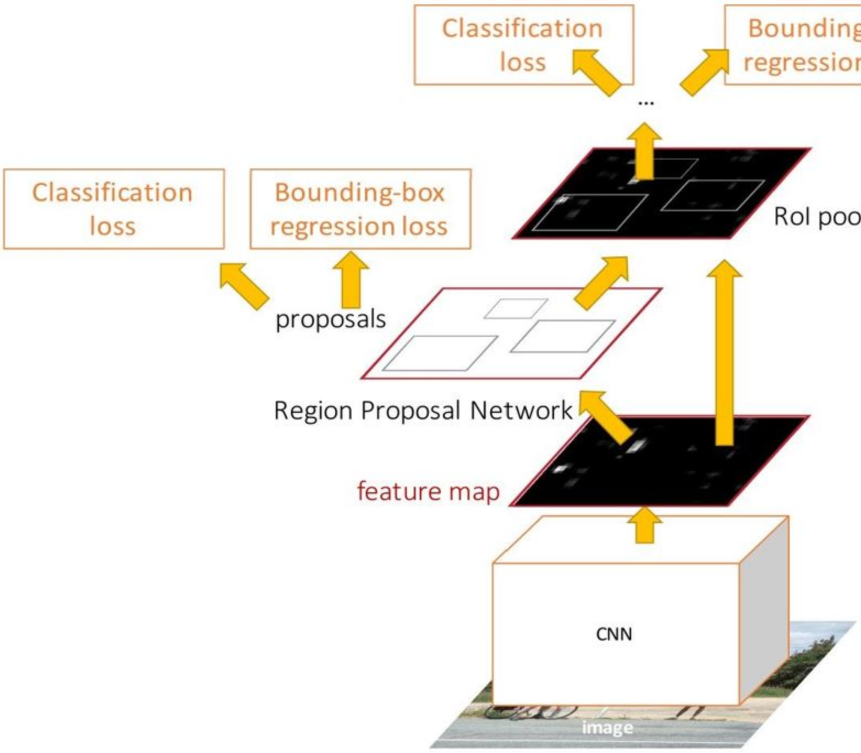


Figure 3.10: Faster R-CNN architecture [59, 39].

2007 [16] dataset using as a convolutional backbone VGG16 [65] demonstrating the validity of the approach also in the case of deep convolutional backbones.

Faster R-CNN introduces the RPN – Region Proposal Network – to optimize the speed of region proposals generation that, previously based on Selective Search, had become the main bottleneck of Fast R-CNN. RPN is a Fully Convolutional Network (FCN) that generates cost-free region proposals through convolutional features sharing with the on-top detector. This can be optimized through multi-task loss on end-to-end training and predicts region proposals with a variety of scales and aspect ratios using the concept of anchor boxes. In particular, to generate region proposals, a sliding window on top of the convolutional feature map output of the last backbone convolutional layer is used. The sliding window is connected to the fully connected layers used respectively for bounding box regression and class-generic object classification that predict $4k$ boxes and $2k$ scores (object vs non-object) with k number of anchors boxes. Subsequently, the non maxima suppression is used to remove boxes that overlap with other boxes that exhibit higher scores. The remaining regions are then propagated through the class-specific detection phase that exploits the original Fast R-CNN detector. The training phase, although end-to-end, optimizes separately the RPN and the subsequent detector or jointly in an approximate way in the case a fine-tuning is performed with the underlying convolutional backbone.

Ablation Studies:

Ablation studies emphasize above all the importance of the RPN. The preferred number of scales and aspect ratios is determined through grid search by opting for 3 distinct scales $\{128^2, 256^2, 512^2\}$ and aspect ratios {2:1, 1:1, 1:2}. The scalability of the approach is underlined in case the backbone is constituted by a very deep convolutional neural network. The validity of the approach *two-stage proposals and detection* is confirmed with respect to the case *single stage detection* where the RPN is used to make class-specific inferences arguing in favor of the double-stage case for faster inferences and better accuracy. Finally, further studies

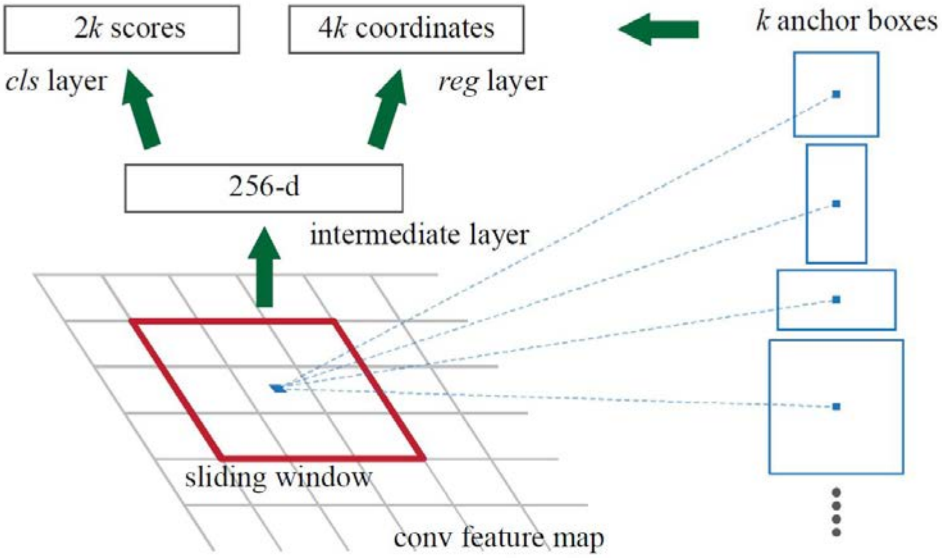


Figure 3.11: Region proposal network [59].

analyze the variation of recall when the IoU varies given a fixed number of region proposals obtaining 0.7 as a good threshold for the post-RPN non maxima suppression.

3.2.5 Mask R-CNN

Mask R-CNN [30] is published in 2017 through a FAIR – Facebook AI Research – project as a framework for instance segmentation. This extends the original features of Faster R-CNN by introducing an inference branch for instance segmentation while maintaining the performance of the previous architecture running at 5 *fps* on GPU. The flexibility of the framework also allows rapid generalization towards complementary tasks such as human pose estimation.

The architecture of Mask R-CNN is a natural integration of the Faster R-CNN architecture. In particular, an FCN [48] is added as an additional inference branch for the prediction of the masks associated with each ROI identified by the traditional branches of bounding box regression and classification. This simple integration is further optimized with respect to the original FCN and Faster R-CNN structures decoupling mask and class predictions and removing the quantization introduced by ROI Pooling through a pixel-to-pixel alignment. Both of these optimizations proved to be essential for good quality results.

The decoupling between mask and class predictions is a divergence from the traditional FCN [48] structure that couples segmentation and classification together and produces per-pixel multi-class categorization outputs. In particular, although the mask branch can predict K masks for ROI, only the k-th mask is used where k is the class predicted by the classification branch. The decoupling is also applied in the multi task loss which in this case sees the combination of the losses of all the output branches – classification, regression, segmentation – and in particular for the mask branch only of the k-th component. This change avoids the mask classes competition problem found in FCN and is the key to good segmentation results.

Faster R-CNN was not designed for pixel-to-pixel alignment between input and output. This feature, although not a problem for the Faster R-CNN structure, is required by Mask R-CNN in order to align the predictions of the masks with the objects in the scene. In particular as noted in [30] the ROI Pooling layer performs coarse spatial quantization for feature extraction and introduces misalignments between the ROI and the extracted features. To solve this problem, the ROI Align layer is introduced which removes the quantization introduced by the ROI Pooling layer through bilinear interpolation of the ROI bins. This approach is of great impact both in the field

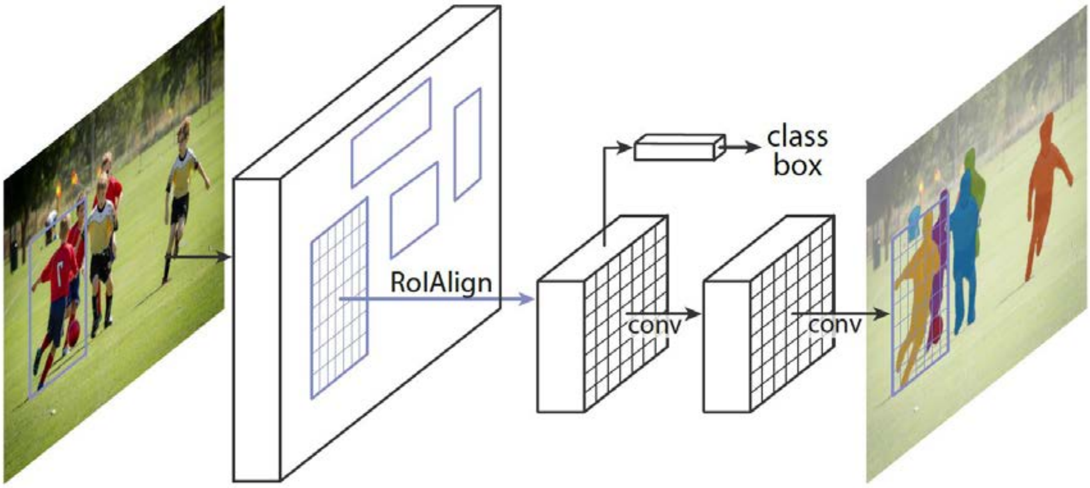


Figure 3.12: Mask R-CNN architecture [30, 39].

of instance segmentation and object detection.

Ablation Studies:

Ablation experiments demonstrate the validity of the Mask R-CNN approach both in the instance segmentation and in the object detection landscapes.

The performances of the framework are studied with different network topologies both for the backbone and the head. For the backbone, the effects of deep network topologies such as ResNet [29] and ResNeXt [77] at depth 50 and 101, as well as the FPN [44] topology are studied. The latter approach is a top down architecture with lateral connections that aims to build high-level semantic features at different scales. For the head, the framework rely on previous works that aims to integrate ResNets [29] and FPN [44] backbones on the original Faster R-CNN architecture [29, 44].

Other studies investigate the effects of multinomial and independent masks as well as class-specific and class-agnostic masks further supporting the mask and class decoupling approach [30]. The changes in performance introduced by the ROI Align layer were examined in detail too, confirming the positive effects introduced by ROI Align both in the instance segmentation and in the object detection field.

Mask R-CNN is a general purpose modular framework [30] that outperforms all existing, single-model entries on every task of the COCO 2016 [46] challenges, including instance segmentation, bounding box object detection and person keypoint detection. The framework can be easily extended to tasks different than instance segmentation as human pose estimation and represents an excellent baseline for research and industrial projects.

3.3 Single-Stage Detectors

Single-stage detectors simplify the object detection problem, originally conceived in a double-stage way by region-based detectors, revisiting it in a regression fashion. Pipeline simplification means faster inference but less accurate results than region-based approaches. These considerations have been confirmed by experimental results in many works [32] but recently further optimizations have been proposed that have challenged previous works [45].

The main single-stage architectures are YOLO – You Only Look Once – revisited in three different versions [58, 56, 57] and SSD – Single Shot multibox Detector [47]. Below we will briefly introduce the main features of the two architectures.

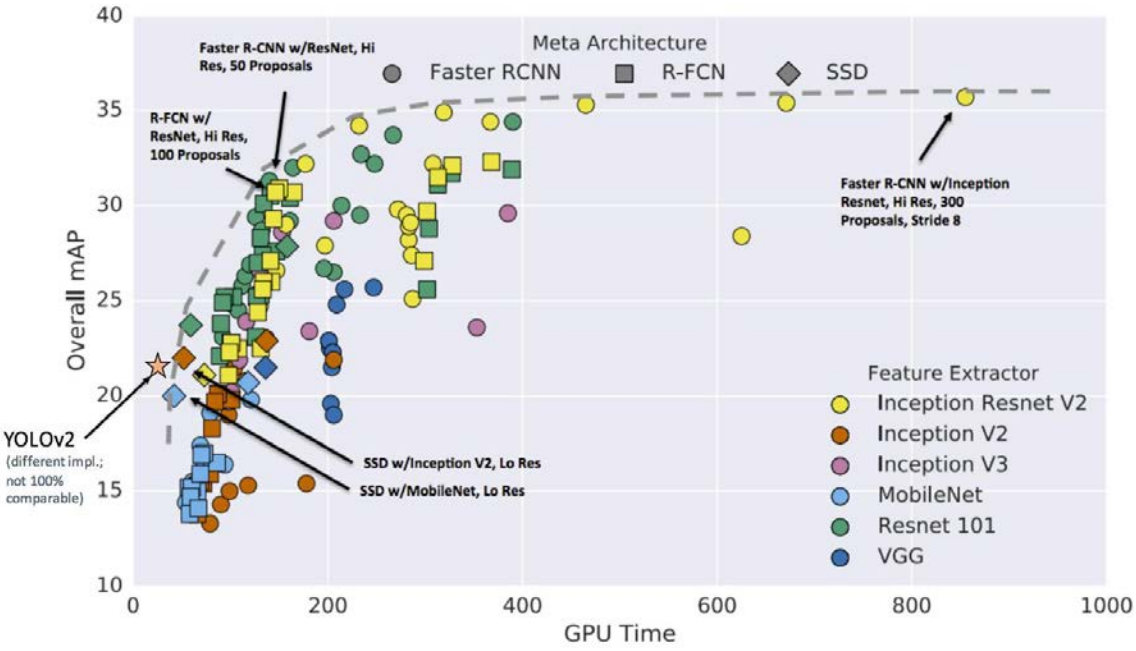
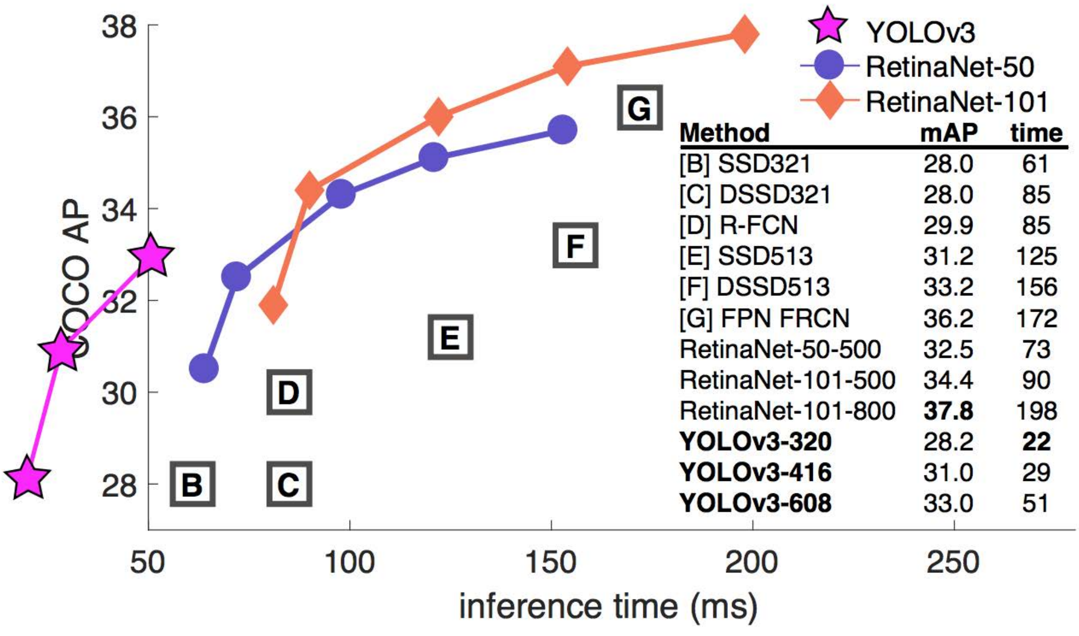


Figure 3.13: Speed over precision on COCO test-dev [32].



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

Figure 3.14: Speed over precision on COCO test-dev [45].

3.3.1 YOLO

YOLO [58, 56, 57] design is based on a convolutional backbone and it enables end-to-end training and real-time speeds while maintaining high accuracy performances. J. Redmon claims to have originally developed a base network running at 45 fps on a Titan X GPU obtaining a mAP of

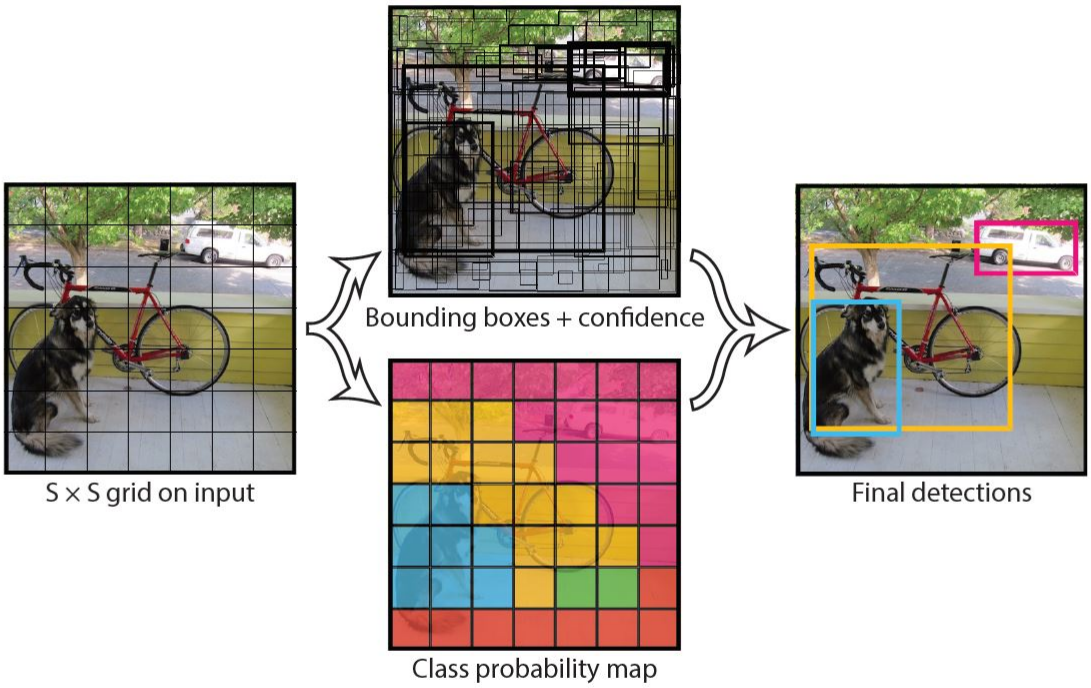


Figure 3.15: YOLO pipeline [58, 56, 57].

63.4% on the PASCAL VOC 2007 [16] dataset and a faster network exceeding 150 fps with a mAP of 52.7% [58]. Subsequent optimization have further developed the architecture by obtaining a trade-off between speed, precision and recall [56, 57].

Differently from region-based approaches, YOLO reasons globally on an image when making predictions. The original inference pipeline is described below.

- resize the input image and divide it into an $S \times S$ grid.
- for each cell predict C class probabilities and B bounding boxes each with 5 predictions: the ROI constraints and a confidence prediction that represents the IoU between the predicted box and any ground truth box.
- multiply the conditional class probabilities and the individual box confidence predictions to encode the class probability map.
- aggregate the final results through non maxima suppression over the class probability map.

Although very elaborate, the original YOLO architecture imposes strong spatial constraints on bounding box predictions that limits the number of nearby objects that the model can predict. In fact, one of the main issues of YOLO is the accurate detection of small objects in groups. This generally involves low recall compared to region-based approaches.

Successive optimizations [56, 57] succeeded in mitigating the negative effects of the original architecture. A trade-off between speed and inference has been obtained by introducing a custom backbone called Darknet, predictions at different scales and aspect ratios, and an alternative loss function which gives greater weight to the contributions of small-sized objects.

3.3.2 SSD

Single Shot multibox Detector [47] is a single-stage detector that is designed to achieve real-time performance and high accuracy inferences. The authors claim to have developed an architecture

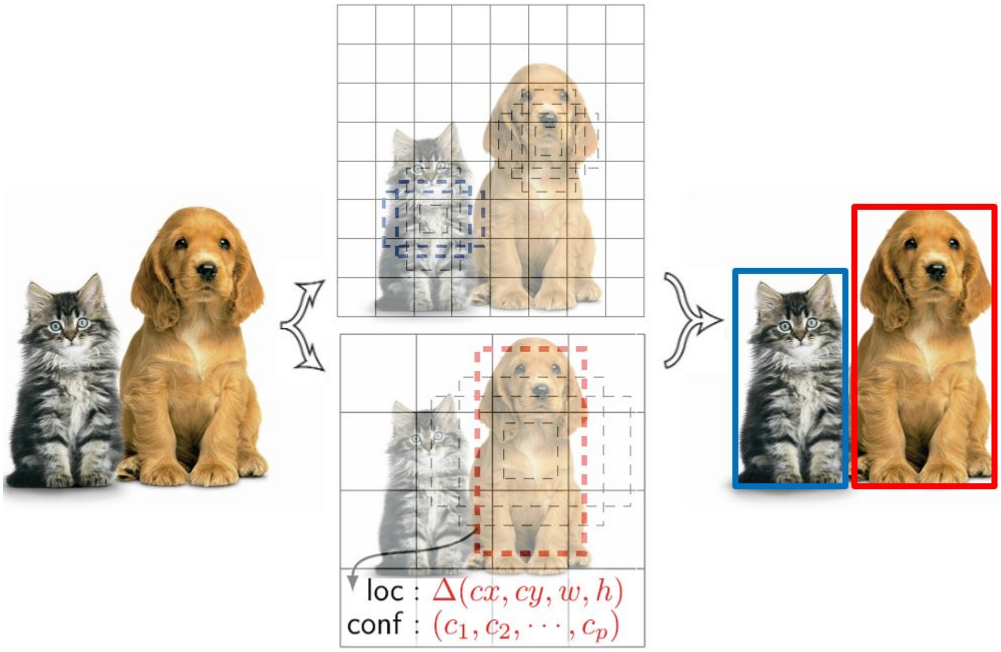

loc : $\Delta(cx, cy, w, h)$
conf : $(c_1, c_2, \cdots, c_p)$

Figure 3.16: SSD Pipeline [47].

capable of achieving a frame rate of 59 fps on a Titan X GPU while obtaining a mAP of 74.3% on the PASCAL VOC 2007 [16] dataset for 300×300 input images.

The SSD architecture is a structure that reflects a connection point between YOLO and region-based structures such as Faster R-CNN. A brief overview of the inference pipeline follows.

- resize the input image and run the convolutional backbone. After reaching the last convolutional layer of the backbone, an $M \times N \times P$ feature map with P channels is obtained.
- for each cell predict B bounding boxes at different scales and aspect-ratios each with $(4+C)$ predictions: the ROI constraints and C class scores.
- aggregate the final results through non maxima suppression.

Accuracy, recall and inference speed are in this case a compromise between YOLO and the region-based final results. The main issue of SSD as in the YOLO case is in inferences for small objects.

Chapter 4

Design

Recently, automated retail systems have been proposed with cross-cutting approaches in systems integration, process automation and big data analytics. In this context, modern deep learning techniques play a key role in the design of innovative systems according to a data-driven approach. Following are presented all the design phases that have defined the development of the retail shelf analytics system for the Customer. The system model aims to analyze images and data from the Customer's management processes in order to provide business intelligence services in a system integration fashion between Customer and Supplier architectures. Recent developments in the instance segmentation field described in the previous chapters will be exploited in order to design a system for the analysis of products on store shelves.

The chapter is organized as follows. The development of the retail shelf analytics system will be discussed in all the design phases from the requirements analysis to the final implementation of the system architecture. The privacy of the Customer will be maintained by obscuring sensitive information. Software and configuration issues will not be treated, but these have made up much of the work in terms of integration between Customer and Supplier IT systems.

4.1 Requirements Analysis

The following is a standard-compliant requirements analysis based on official documentation. The existing system, functional and non-functional requirements and architectural constraints will be analyzed in detail, while maintaining the privacy of the Supplier and the Customer.

4.1.1 Objectives and Description of the System

The purpose of this project is the design of a Proof Of Concept (POC) that integrates an existing retail system analytics architecture with advanced analytics functionalities based on deep learning. The project is defined in terms of end-to-end integration with the business activities of the Customer. The Customer has a relevant presence in national pharmacies. The operations that affect this business activity are many and aim to propose, through agents and merchandisers, commercial promotions targeting certain topics, such as oral health, intimate wellness, and pain relief. Currently there is an activity on the Customer side that aims to provide business intelligence services through a SAP Customer Relationship Management (CRM) system. This system develops a whole series of KPIs based on data entered by the agents. These data, in the development scenario of our work, aim to describe the set-up activities within the pharmacy chain. Other intelligence activities based on other types of data such as revenue and other economic indicators are currently present but will not be treated for privacy reasons.

The data currently present in the intelligence process considered are structured (database entries) and unstructured (images and text). At each visit that an agent or merchandiser performs in a pharmacy, he/she is required to fill in a questionnaire related to the campaign and/or shoot a sequence of images related to the promotional activity carried out in the pharmacy. The images, in particular, represent the business context objectively and are suitable for intelligence operations to provide indications of strategic importance.

Insight activity by the designed AI system will exploit the analysis of these images to extract structured information by binding them to the business context of the pharmacy. In particular, a neural network [30] will be designed to identify products and objectively intercept information of interest. The output of the network will be further elaborated to calculate specific key performance indicators (KPIs). These KPIs will be specifically designed to provide strategic information on the Customer's business (Section 5.3). In particular, with a view to warning identification, the work carried out aims to determine a series of KPIs and metrics to:

- verify the correctness of the information present in the questionnaires comparing them with the information extracted from the artificial intelligence through the analysis of the images;
- measure how much the Customer's brand is influential on each point of sale, diversifying data by survey context;
- verify visual dispersion of Customer products by analyzing the arrangement of products on each shelf and comparing it with that of other brands.

The combination of analysis, images of the layouts that best represent the various campaigns, and the structured data present in the client's system will allow the marketing/sales department to generate Insights on the exposure methodology based on quantitative data.

4.1.2 Functional Requirements

The functional requirements the system must meet are listed below.

The core processing system must be designed to extract the following information from images.

- class of each product identified.
- bounding box position of each product identified.
- score of each product identified.
- number of pixels of each product identified.
- number of shelves.
- shelf type as defined in specific enterprise document.
- promotional campaign type as defined in specific enterprise document.
- cluster type as defined in specific enterprise document.

The system must provide a database to collect the following information.

- information extracted from the images, as listed above.
- client systems activities information about the questionnaires and other useful information in the survey domain.

The system must make the following information usable through visual analytics tools.

- KPIs verification matching elaborations results with questionnaires information.
- customer brand influence over competition through different AI-based aggregation metrics.
- dispersion of Customer products on individual shelves against competition.

Entity	Description	Attributes
Task	Single activity of the Customer system that in our domain of interest, it constitutes the set of activities of Merchandisers and Agents for the operations on in-store products	Task ID, SubTask ID, Task Description, Task Date
Task Type	Customer's Task Type	TaskType ID, TaskType Description
Client	Pharmacies selling products of Customer	Client ID, Client Name, Location, Province
Survey	Questionnaire related to Customer activities	Survey ID, Survey Description
Question	Question in the Survey	Question ID, Question Description
Answer	Question's Answer	Answer ID, Answer Description
Picture	Image taken in the single Task	Picture ID, Picture Path
Metrics	Metrics calculated on the single Picture	Shelves Number, Cluster, Shelf Type, Promotional Campaign
Object	Object identified in the Picture	Object ID, Class, Score, Top, Left, Width, Height, Pixels Number, Shelf

Table 4.1: Data dictionary – entity table.

4.1.3 Non-Functional Requirements

The non-functional requirements are described below.

- the system must perform the operations completely into the Supplier company's systems replicating Customer data if necessary.
- the system must be designed to provide a daily computation run to process the new data entered into the system in batch.

4.1.4 Constraints

The main constraints of the system follow.

- the system must be delivered in six months.
- the core processing engine must be based on TensorFlow.
- the system must be integrated into the Supplier's SAP systems.
- the results produced by the system must be accessible in the cloud through SAC – SAP Analytics Cloud.

4.2 Data Logic

The following is a brief summary of the data layer on which the existing system is integrated. As agreed, only aspects related to the domain of interest will be treated, maintaining anonymity and confidentiality on the overall system.

4.2.1 Entity-Relationship Schema

The ER Schema represents the mini-world of interest. We can define in detail the key entities that constitute our model and how they interact with one another.

In particular, we can observe that the main entity is the Task. This constitutes the set of activities of the Customer system that in our domain of interest are related to the activities of preparation and maintenance of store shelves by agents and merchandisers. The Task is associated with a Task Type, an end Client and a set of Pictures and Surveys. We have emphasized with

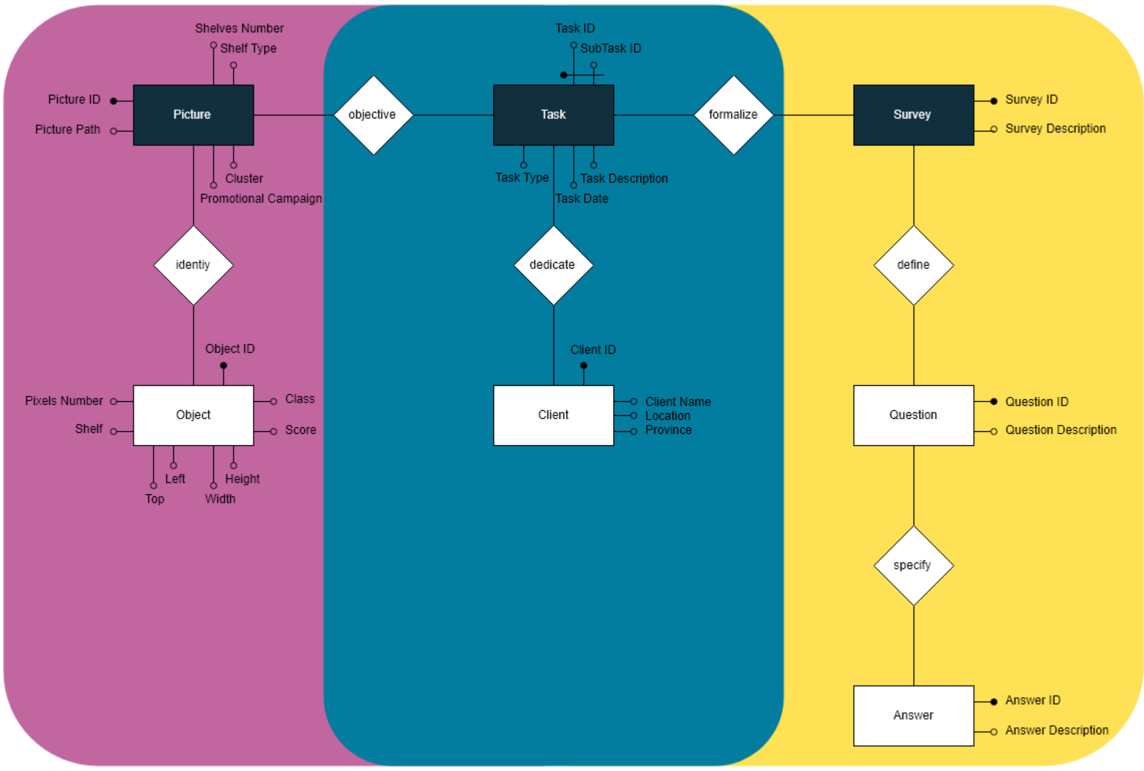


Figure 4.1: Entity-Relationship schema.

different shades the entities and relationships that respectively involve the Surveys and the Pictures associated with each Task. The Pictures are characterized by a set of attributes that specify the KPIs identified by the AI algorithm in terms of Shelf Type, Promotional Campaign, Cluster and Shelf Number as previously established. A further entity connected to these constitutes the set of Objects identified by the algorithm in each Picture, characterized in terms of Class, Score, Top, Left, Width, Height, Pixels Number and Shelf. The Survey is linked to the Questions that define it while the latter are associated with the Answers.

4.3 Presentation Logic

The development of the presentation logic starts from the requirements analysis and defines the functionality of the graphic objects and how these interact to guarantee the services that the system intends to make available to the Customer. The front-end follows with the mock-ups.

4.3.1 Mock-Ups

Mock-ups map user interaction in the various application screens. Specific models have been constructed to define the services associated with each screen. Below are just some of the basic models, while the final models will be discussed in more detail in Section 5.3. The proposed mock-ups follow a basic template consisting of a page filter and the associated graphic objects. Using the page filter it is possible to filter the results for the specific KPIs selected. The user interface is dynamically transformed according to the selected KPIs proposing customized display modes in real time according to the needs of the Customer.

The first mock-up proposes the representation of the presence of Customer products in pharmacies according to various aggregation methods. Presence will be defined in terms of the number of products and the number of pixels. The aggregation methods used will instead investigate the

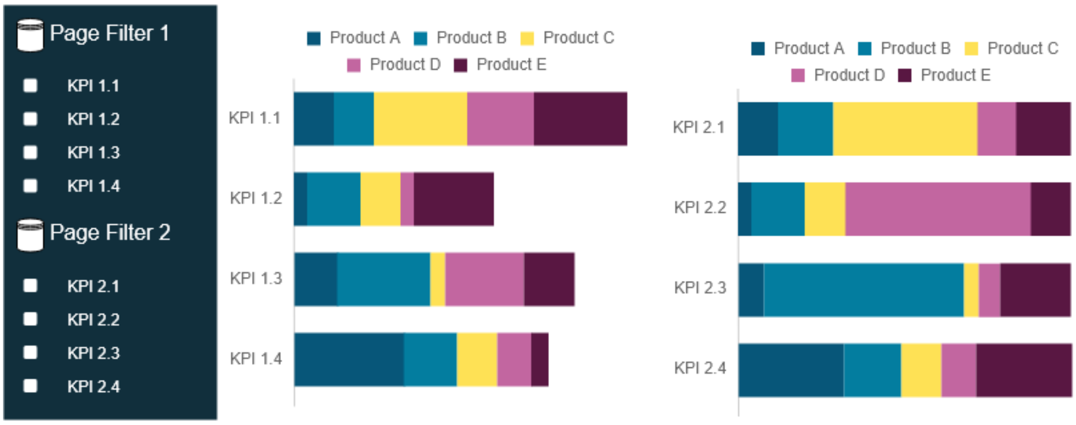


Figure 4.2: Mock-up – Product insights in stores.

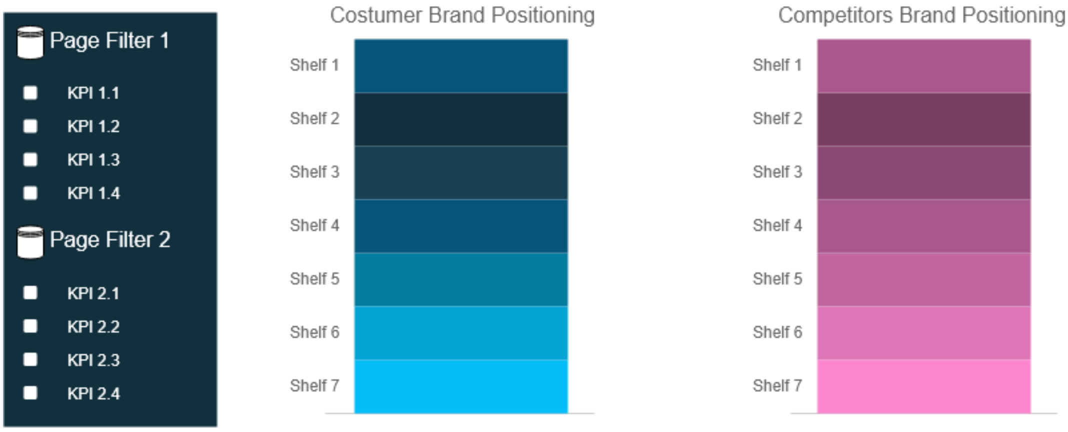


Figure 4.3: Mock-up – Product positioning on shelves.

previously defined metrics in terms of Installation Type, Promotional Campaign and Cluster according to previously approved document. The results will be proposed through stacked bar plots and it will be possible to analyze in detail each single component with a simple click. The graphs will be dynamically transformed depending on the KPIs selected in the page filter.

The second mock-up represents the positioning of products on individual shelves in the complex of pharmacies analyzed. This allows to define in terms of Installation Type, Promotional Campaign, Cluster and in the global complex the influence of the Customer's brand with respect to competitors through an heat map representation. The variation of colors in the heat map allows to analytically visualize in which shelves the various products are concentrated and check if this complies with the official agreements. Also in this case the results will be displayed dynamically depending on the status of the page filter.

4.4 Business Logic

A privacy-compliant description of the business logic follows. A brief overview of the Supplier and Customer IT systems will be presented. Then, the algorithmic implementation will be described.

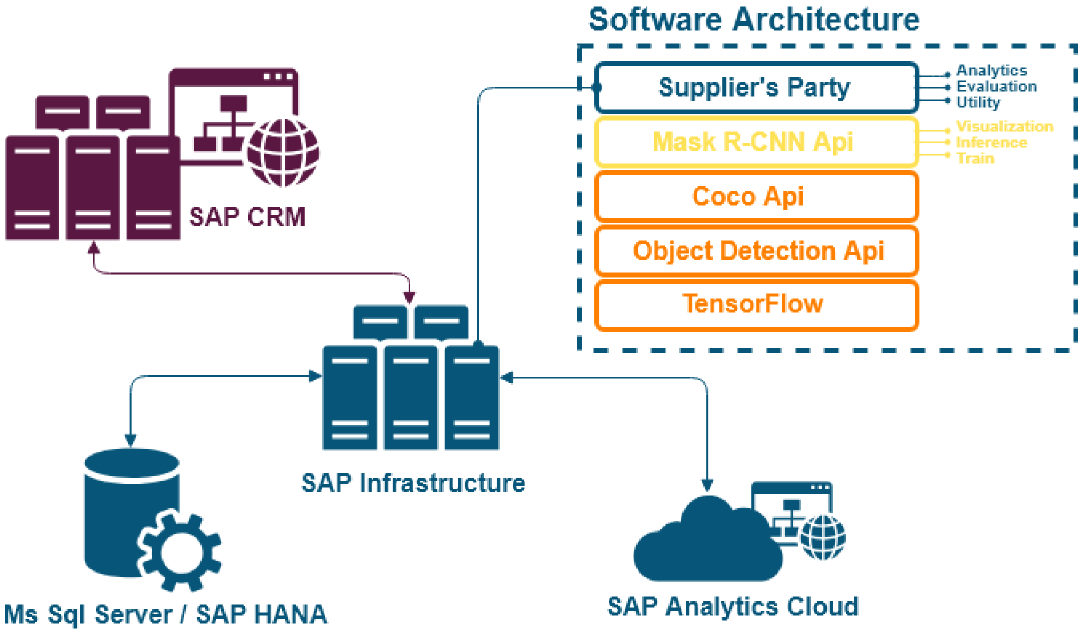


Figure 4.4: Business logic – Model view controller.

4.4.1 System Overview

The main entities in the system are the Customer and Supplier IT systems. These are based on SAP software products that expose multiple services through an MVC logic.

The Customer infrastructure is based on SAP CRM, which allows to manage the user experience at the sales, marketing, customer service and digital commerce activities levels. This product specifically tracks the way in which businesses interact with customers with analytics, e-commerce and social media tools. This architecture hosts the intelligence data in the form of KPIs, questionnaires and images that will be processed by the Supplier system. Data will be processed daily by the Supplier system in a synchronized way through a data flow between the two systems.

The Supplier system is also integrated in a SAP solution. In particular, in our domain of interest while originally it used two core products of the suite respectively for database management systems and analytics, actually due to some performance constraints on the Supplier server side, the database relies on Microsoft technology. Intermediate solutions based on MySQL have also been developed but being based on open-source technology they have not been formally approved by Customer and it has been necessary to redesign the database according to a commercial alternative. Because of this limitation, the data layer is currently based on Microsoft SQL Server but according to the original proposal it is already planned to migrate it to a SAP HANA system when the Supplier systems will return to the original configuration. This system is particularly performing as it consists of an in-memory database optimized for high performance computing. At the application level, instead, SAP Analytics Cloud (SAC) is used with a front-end logic to provide interactive results in cloud based on the associated data layer. The product provides finance, HR, operations, sales and marketing solutions and is particularly useful as a centralized storytelling tool.

The software architecture is integrated into IT Supplier systems and is based on TensorFlow [51]. In particular, there is an open-source library stack on which the Supplier software leans. Object detection APIs [32] are used as an abstraction layer to create deep learning models able to locate and identify multiple objects in images. The COCO APIs [46] are instead a support package to load, parse, and visualize COCO-like annotations to images. These are a de facto standard because the COCO dataset is used for the most popular challenges in the field of object detection, instance segmentation and Keypoints Detection. On top of COCO APIs, we integrated an open-source implementation [1] of the Mask R-CNN [30] algorithm. The code has been released

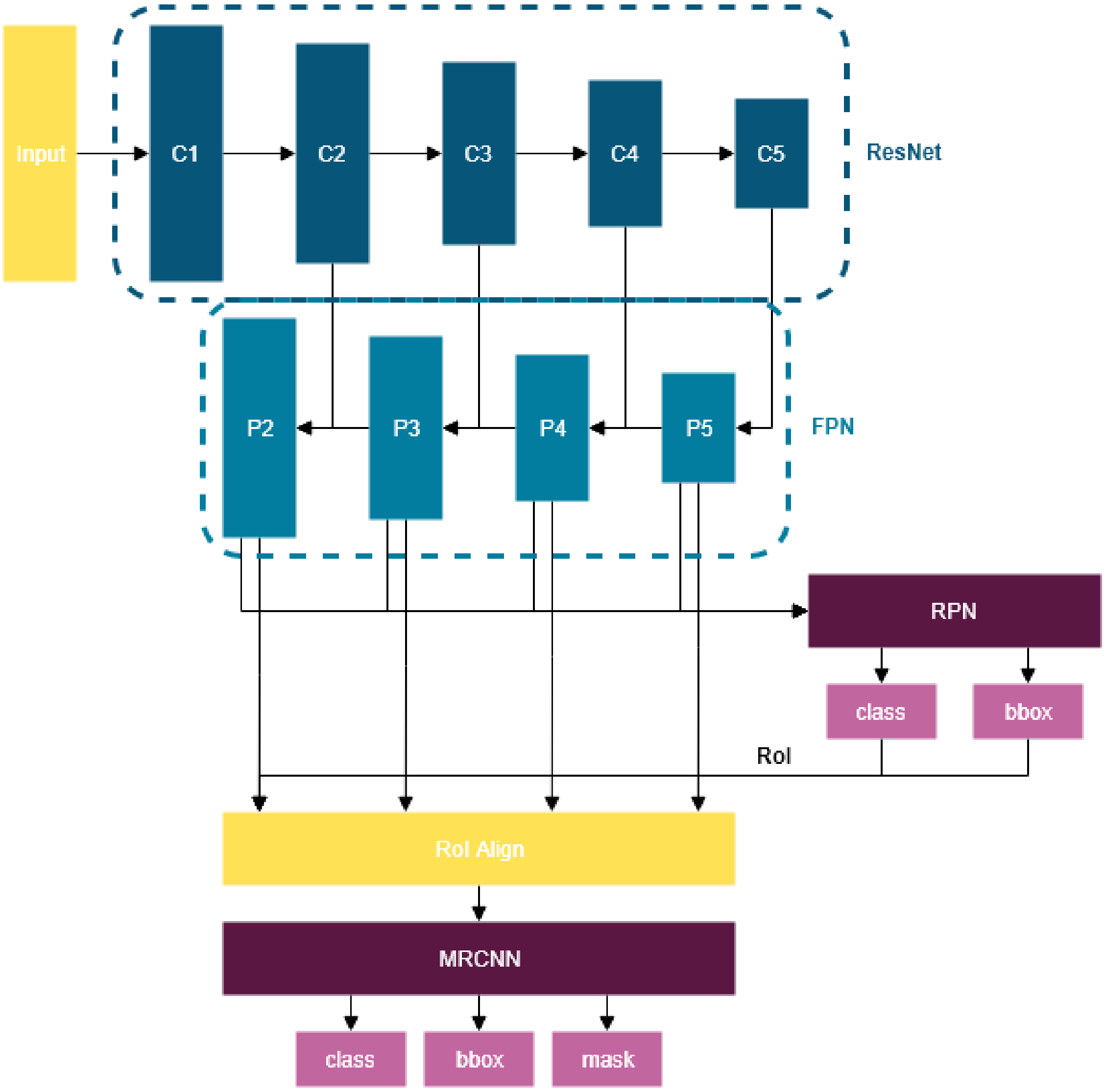


Figure 4.5: Customized Mask R-CNN architecture.

by Matterport Inc, a company that provides a cloud based platform that enables users to create and share 3D and virtual reality models of real-world spaces as well as a proprietary 3D camera for the acquisition of spatial data. The algorithmic implementation of Mask R-CNN is based on TensorFlow and in particular Keras for the high level development of the neural network. This provides utilities for network training and inference as well as for displaying results. The Supplier software is designed as the last level of customization of Mask R-CNN as well as an analytics tool based on the predictions of the algorithm. This includes design, model selection and analytics features as well as additional support capabilities. The analytics features are designed according to an official enterprise document to extrapolate the metrics previously described. The evaluation features allow to perform model selection based on metrics such as precision, recall and F-Measure calculated on the inferences of the algorithm on the test set. The result sets are further elaborated to determine if two candidate models exhibit equivalent behavior using the Wilcoxon signed-rank test. These last elaborations will be described in detail in the Chapter 5, devoted to experimental results. The support functions, instead, provide utilities for the management of the database system as well as for the management of the daily batch processing.

The designed system allows to choose whether to perform the inferences with Mask R-CNN,

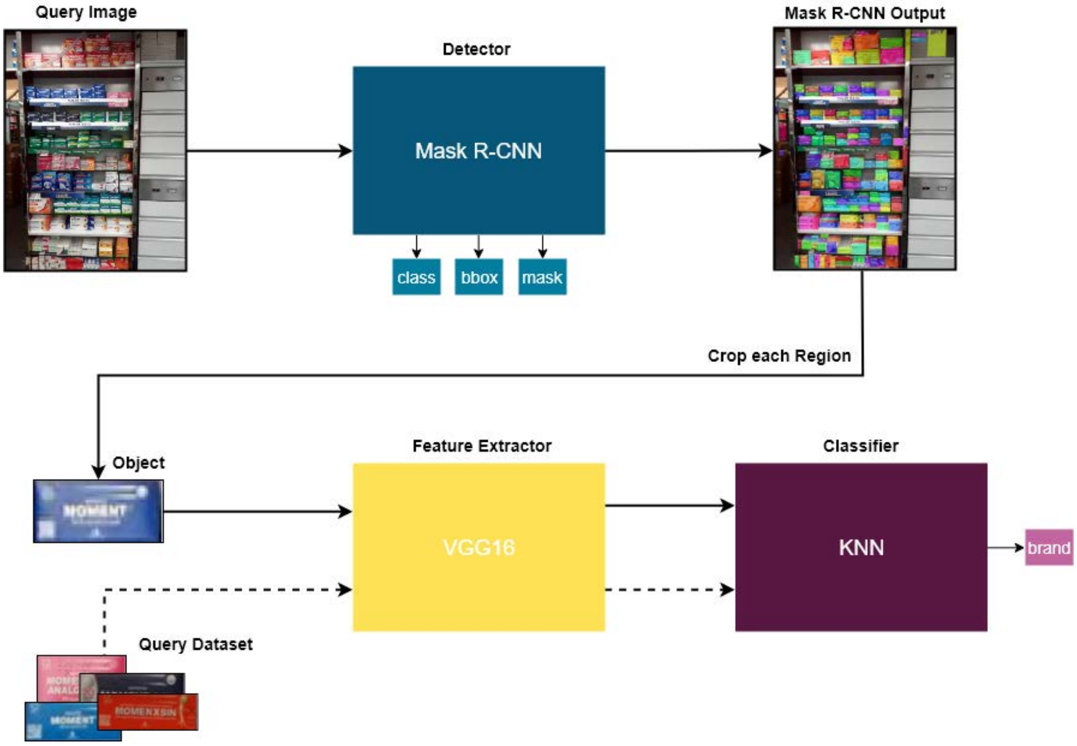


Figure 4.6: Deep learning brand recognition pipeline.

to recognize generic or specific products superclasses, or to use a double-stage brand detector to obtain a greater level of detail. Mask R-CNN [30] has been customized to the task of interest through an optimized network configuration. Specific experimental tests have been designed to fine-tune the network and will be described in the related sections. The final network configuration includes a ResNet-50 [29] convolutional backbone with an FPN [44] topology. Many of the RPN [59] configuration parameters such as anchor scales and anchor ratios as well as the non-maxima suppression threshold have been determined by model selection or visual analytics in order to maximize precision, recall or F-measure calculated over the whole test set. On top of Mask R-CNN, a brand detector similar in spirit to the one of Tonioni [71] has been implemented. This exploits the KNN [12] algorithm to classify the objects identified by Mask R-CNN at the brand level through a feature extractor based on VGG16 [65]. In particular, the regions extracted by Mask R-CNN are sent to VGG16 which extracts the features. These are compared with the products features previously extracted by VGG16 in a query dataset through the KNN algorithm. This solution is currently in prototype form but some results will be presented in the Chapter 5. In particular, the detector allows us to abstract the retail context and classify different products without having to own a specialized dataset with the annotations of the various classes. This result can indeed be obtained by having a unique dataset with annotations related to the recognition of a generic product superclass accompanied by a query dataset of reduced dimensions of images of real products from the Customer catalog to be recognized. Given the reduced number of images in the Ad-Hoc dataset, this approach is currently the only one that can classify objects at the brand level in the Customer business domain.

4.5 Dataset Creation

The creation of the dataset is a central problem in supervised learning and in particular in the training of modern neural networks [39, 55]. Currently it is possible to take advantage of different

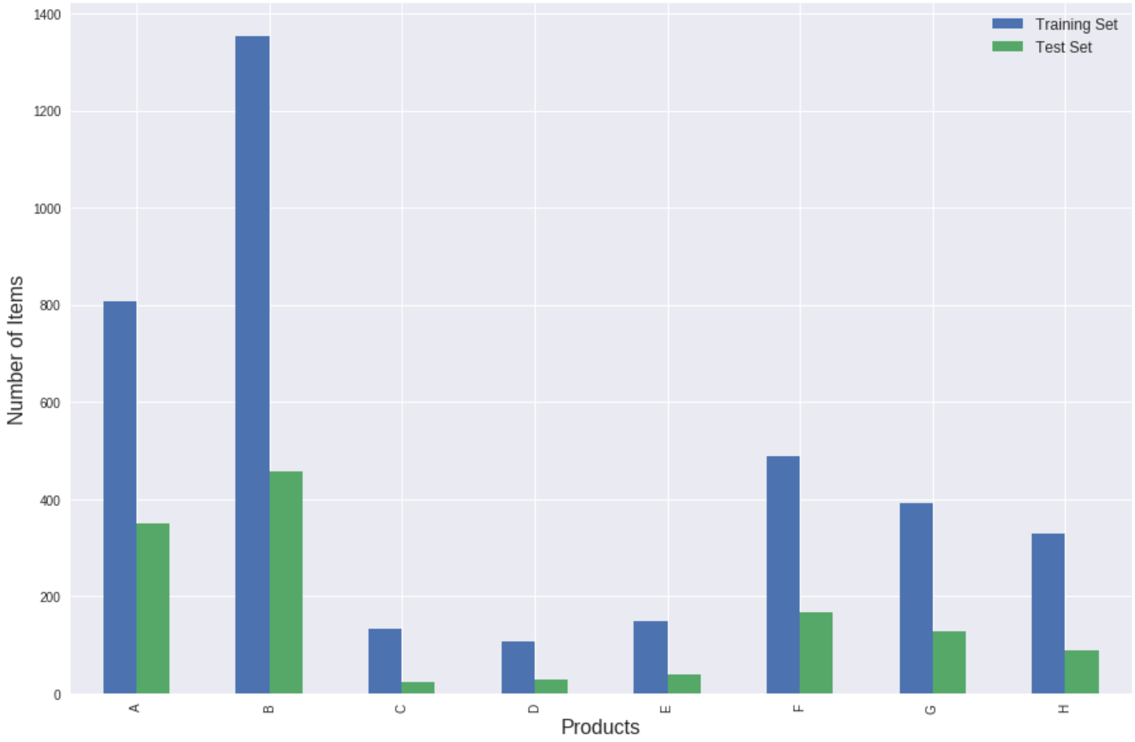


Figure 4.7: Ad-Hoc dataset customer product histogram.

strategies to reduce the number of necessary annotations. Among these, transfer learning [78] is undoubtedly the main development tool that allows to reduce both annotations and training time considerably. However, especially for neural networks the number of necessary annotations remains high and other directions such as weak supervision, data augmentation and synthetic data constitute a very active research field to further optimize training performance.

In parallel to the study of modern techniques able to optimize training, new services and specialized companies in dataset creation are developed on the business side. Amazon Mechanical Turk (MTurk) [6], in particular, is a marketplace for strategic outsourcing of business processes based on task crowdsourcing on a distributed workforce around the world. This allows to delegate simple time-consuming tasks to a specialized workforce, considerably reducing time with low costs. MTurk is one of the main tools used by data scientists to obtain annotations on training data in a short time. The annotations are then evaluated on job quality metrics in order to rule out any errors due to possible workers distraction [37].

The creation of the dataset in the Customer scenario has been particularly critical given the impossibility to use services such as MTurk for privacy issues on Customer data. In particular, the construction of the dataset has been an iterative process where a lot of energy has been invested and several experimental tests have been carried out to evaluate various alternatives (Subsection 5.2.2). Attempts have been made to automate the annotation process but proved to be inadequate. In particular, as described in Section 1.1, it was not possible to use the annotations from other retail datasets to obtain a suboptimal solution since the latter targeted different scenarios. Other attempts at automation have been implemented in terms of weak supervision, but even these have not proved to be optimal. The Ad-Hoc dataset is therefore currently the only resource available for training the neural networks on the Customer business processes.

In addition to the automation tests (Chapter 5) of the dataset annotation process, the main activity, given the previous premises and the peculiarity of the type of application, involved the manual annotation of the images coming from the Customer's business processes. This has been a great investment both in terms of time and energies given that to label all the pixels of every single image, considering the high number of items present in the images (157 in mean) and the type of annotation (pixel-wise) an average of 30 minutes has been spent on each image. Moreover, other episodes related to the main consultancy activity with Customer have further delayed the process given the impossibility of initially having the requested images. While respecting the privacy restrictions with Supplier and Customer, and wishing to maintain confidentiality, these sensitive details will not be treated, but giving an hint, it is possible to say that a lot of time has been invested also in the relations with the different Customer entities in order to obtain good quality images. In fact, since initially the images that Customer had provided were not considered adequate given the poor quality as well as sometimes the absence of the pharmaceutical background, new ones have been required to meet these limitations.

Currently the Customer dataset consists of both types: low resolution images (not labeled) and high resolution images (labeled). The former are the most consistent part of the dataset but given the previous limitations have not yet been directly used. Some of the experiments presented in Chapter 5 will try to define a trade-off between the performance of the algorithm inferences and the quality of the images used for the training and the test. As for the good quality image pool, these are currently the only ones used for the training of Mask R-CNN [30]. Currently, as anticipated in Chapter 1, the main dataset (good quality images) consists of 410 images. For model selection purposes this dataset has been further divided into a 300-image training set and a test set with 110 images. Considering the total size of the good quality image pool conferred by Customer (550), it has been decided to keep the test set of 110 images to make all the results comparable as a result of possible future extensions of the training set. The latter, currently composed of 300 images, may in future be extended by the remaining 140 non-annotated images for a total of 440 images. As we can see from Figure 4.7, the overall dataset is made up of 8 different classes of Customer products to which a class of competitors products (not shown) is added. The distribution of the products is clearly unbalanced with respect to the products A and B which constitute the core of the Customer business as we will discuss in Section 5.3. However, although this imbalance does not allow to obtain statistically significant results for each case, all the products have been considered equally in the final analysis.

In addition to manual labeling, given the great amount of work necessary to annotate all the images, the design phase of the dataset also involved other activities for selecting the annotation tools that are preferable for a fast construction of the dataset. The annotation tools available online are various, some available open-source and others based on proprietary license. As for the specific tools that can annotate the main applications of object detection presented in Chapter 3, these are able to label images for tasks such as image classification, object detection, semantic segmentation, instance segmentation and others. Furthermore, the variability of the tool is also given by the different methods able to obtain the same annotations output. In particular, as regards the landscape of the semantic segmentation and instance segmentation given the peculiarity of the fields, there are currently 3 different types of tool available. Specifically for pixel-wise labeling, there are tools that take advantage of geometric segmentations [15], tools that take advantage of the watershed algorithm [3] and more modern tools that use neural network models [2] to return the final annotations once again in the form of geometric segmentations. Given the peculiarity of the business domain studied and being that all the products for which Customer has requested identification are based on a rectangular form, among all the available open-source tools, VIA [15] has been chosen to obtain instance segmentation annotations starting from geometric images segmentations. Having made tests (not reported) to monitor the annotation time with each tool available, VIA has been chosen using as a selection criterion as well as the quality of the annotations, the reduced effort in terms of time compared to the remaining tools. In particular, although the other tools allowed to obtain very precise annotations, much effort have been necessary in the final manual remodeling of the annotations starting from the inferences of the core algorithms. As for the tool based on neural networks, although it has been very promising, it presented the same problem due to the manual remodeling of the annotations following the inferences of the algorithm. In this case, higher performances could be obtained following a fine-tuning of the model based on Recurrent Neural Networks (RNN) but not having at the time available an adequate number of images initially for model training, the VIA tool has been preferred.

Chapter 5

Experimental Results

Experimental evaluation is the main tool for model performance benchmarking. This is used in the main object detection international challenges to reward the most innovative solutions. Experimental evaluation makes it possible both to confirm theoretical properties and to give an experimental value to properties for whom it is not possible to reach a formal demonstration. The most innovative research projects exploit experimental evaluation techniques to improve the state of the art and reach higher performance standards.

This chapter will cover the following topics. First of all, an introduction to the main object detection experimental evaluation techniques with reference to the Customer use-case will be presented. Then, the experimental tests that helped determine the final network architecture will be discussed. These will cover the attempts to automate the manual labeling process of the dataset, the fine-tuning of the network and the final configurations. Other tests in line with the optimization of the training time and the Customer data analytics investigate the effects of the training configurations and the resolution of images on final performances. Finally, the main business intelligence and visualization results obtained on the Customer data will be presented.

5.1 Experimental Evaluation

The basis of the experimental evaluation were developed in the Information Retrieval field through the famous Cranfield paradigm, in the years 1958-1966 [8, 9]. This discipline studies the effectiveness measures and the experimental tests that allow to determine how a system behaves according to the different information retrieval situations in which it could be used [27]. With these tests it is therefore possible to evaluate the performances of the models and to make a selection to determine the best solution in the development scenario. In particular, starting from a corpus of items called *p*ool it is possible to evaluate the output of the retrieval system called *r*un, on the basis of the contingency table through the following measures.

Precision. Indicates how many relevant documents have been retrieved in response to a query compared to all those retrieved:

$$Precision = \frac{TP}{TP + FP}.$$

Recall. Indicates how many relevant documents have been retrieved in response to a query compared to all the relevant ones in the collection:

$$Recall = \frac{TP}{TP + FN}.$$

F-Measure (F_1). The harmonic mean of Precision and Recall:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}.$$

	Relevant	Not Relevant
Retrieved	True Positive – TP	False Positive – FP
Not Retrieved	False Negative – FN	True Negative – TN

Table 5.1: Contingency table.

Based on such measures, it is possible to define further metrics.

Average Precision (AP). Indicates a global precision measure of the run in output from the system. It can be calculated as the average of all the precision values defined on the run or as the average of the precision values interpolated to fixed cut-off levels:

$$AP = \frac{1}{N} \cdot \sum_{i=0}^{N} Precision_i.$$

Mean Average Precision (mAP). It is the average on all APs calculated for each topic in the collection:

$$mAP = \frac{1}{|T|} \cdot \sum_{t \in T} AP_t.$$

The metrics defined above are the main model evaluation tool in the information retrieval field but many other metrics can be used concurrently with the former to further deepen the analysis towards adjacent survey areas. These are generally contextualized in a specific evaluation pipeline depending on the considered use-case. Below we will discuss the main model evaluation pipeline in the object detection field.

5.1.1 Experimental Evaluation in Object Detection

Experimental evaluation can be easily extended in other investigation dimensions. In particular, in the object detection field, it is also necessary to contextualize these metrics based on the predictions of the models [39]. Precision, Recall, F-Measure and aggregated metrics are in fact defined in compliance with a further metric called IoU – Intersection over Union – of how much two bounding boxes or masks are overlapping. Specifically, in most of the cases in literature, the best model is selected through an average precision measure according to a previously defined intersection over union threshold – $mAP@IoU$.

Intersection over Union (IoU). Given a ground truth bounding box B_{gt} and a predicted one B_p the IoU indicates the ratio between the overlapping area of the boxes and the sum of the areas of the boxes:

$$IoU = \frac{area(Bp \cap B_{gt})}{area(B_p \cup B_{gt})}.$$

The effects of the latter threshold in particular are studied differently depending on the context of international challenge considered (e.g., PASCAL VOC [16], ILSVRC [62], Ms COCO [46]) since different challenges have different evaluation rules. In particular, while for PASCAL VOC considers as an evaluation metric $mAP@IoU = 0.5$, Ms COCO selects $mAP@IoU = [0.5 : 0.95]$, where the latter is an aggregated metric on different degrees of intersection over union. The evaluation steps of a classic object detection challenge are therefore the following for every submitted model.

1. Model predictions are computed over a reference test set.

2. Predictions are sorted by decreasing confidence.

3. Predictions are assigned pairwise to the ground-truth items in the test set according to the fixed Intersection over Union threshold.

4. Given the IoU, AP is computed for each class.

5. Mean average precision is computed by averaging the APs of all classes on a target IoU.

6. Further context-specific metrics are eventually computed.

With reference to the last point, in the Customer use-case other metrics have been developed in order to obtain a more detailed evaluation on target investigation areas. These include:

(Mean) Average Recall (mAR). Indicates a global recall measure of the run in output from the system. It can be calculated as the average of all the maximum recalls obtained for each class:

$$AR = \max_{i=0}^{N} Recall_i \qquad\qquad mAR = \frac{1}{|T|} \cdot \sum_{t \in T} AR_t.$$

(Mean) Average F-Measure (mAF_1) Indicates a global F-Measure of the run in output from the system. It can be calculated as the F-Measure of the aggregated precision and recall metrics:

$$mAF_1 = 2 \cdot \frac{mAP \cdot mAR}{mAP + mAR}.$$

Wilcoxon signed rank. In terms of model selection, the Wilcoxon signed rank hypothesis test is an excellent tool for evaluating model performance. This is used as a non-parametric hypothesis test to compare the series of measurements in output from two different systems. In particular, this test is used in this context to determine whether or not model predictions come from the same probability distribution.

Based on these metrics, it is possible to define the following extended evaluation pipeline in the Customer use-case.

1. Model predictions are computed over a reference test set.

2. Mean average precision is computed by averaging the APs of all classes on a target IoU.

3. Mean average recall and mean average F-Measure are computed by averaging over all classes on a target IoU.

4. A final measure for $mAP@@IoU = [0.5 : 0.95]$, $mAR@IoU = [0.5 : 0.95]$ and $mAF_1@IoU = [0.5 : 0.95]$ is computed by aggregating the previous measures on a target IoU threshold set.

5. Wilcoxon Signed Rank test is used to evaluate the submitted model with a target one in order to ensure that the predictions do not come from the same distribution.

In the following sections, the experimental evaluation of the models in the Customer scenario starting from the previously defined pipeline will therefore integrate the metrics built ad-hoc to fine-tune the network architecture and determine the best model.

5.2 Experimental Tests

The experimental results are based on a specialized evaluation pipeline. This integrates the traditional metrics calculated for the evaluation of object detection models with measures defined on the target survey areas. The main results will be analyzed in detail below.

		Detection			Segmentation		
Configuration	Time	mAP	mAR	mAF	mAP	mAR	mAF
Square@1024	4.7 h	52.1%	35.8%	42.2%	52.1%	36.1%	42.6%
Square@1792	13.0 h	57.4%	37.1%	45.0%	58.2%	38.2%	46.0%
Crop@512	2.8 h	52.6%	41.3%	46.1%	53.7%	42.5%	47.3%
Crop@1024	**4.6 h**	**57.3%**	**40.3%**	**47.1%**	**58.8%**	**41.9%**	**48.8%**

Table 5.2: Effects of different training configurations on final performances.

5.2.1 Training Configuration and Image Resolution

One of the main limitations of any system based on neural networks is training time [39]. This is a very strong constraint for experimental tests that have to explore a vast design space, which is also the case for the system described in the Chapter 2. Training time optimization is currently an active research area and some progress on the hardware and software side has been made but a lot of work is still to be done. One of the main limitations to training time in the Customer use-case in particular is due to the size of the images. The problem has been much studied in the literature and in particular the use of high-resolution images during training is to be evaluated carefully, given that training times increase considerably. One of the objectives of this section is to investigate the effects of different training configurations in order to determine a preferential one for the network training.

Another aspect to consider for the training of neural networks, is the size of the training set [39]. Taking as reference what is described in Section 2.3.3, and wanting to exploit the potential of transfer learning [78] all following tests use the same neural network training configuration. In particular, as a training mode, a fine tuning has been chosen starting from a pre-trained model of Mask R-CNN [30] with the weight of the last layer initially assigned randomly. This choice was made considering respectively the reduced dimensions of the Ad-Hoc dataset and the little similarity of the latter with the one used to train the pre-trained Mask R-CNN model (Microsoft COCO [46]).

Since the dataset used for training (Section 4.5) is composed entirely of high-resolution images with a view to optimization, we have monitored various training configurations. In particular, we compared the performance of various configurations to obtain a trade-off between accuracy of inferences and training time while maintaining the comparability of the models. In the Matterport's implementation [1] of the Mask R-CNN algorithm [30] there are two main training modes: Square and Crop. In the first mode, the networks are trained by augmenting the input images so that they become square. In this case, an image is built so that the long side coincides with the original dimension and the short side reaches the same dimension by zero padding. The Crop mode, on the other hand, requires that the network receives in input a random crop of the original image with square dimensions. All images input to the network must have sizes multiple of 64 to meet the requirement placed by the FPN [44]. Furthermore, all images must have the same dimensions within the same batch in training mode. The latter is an hyperparameter that regulates the number of training samples used for the network training at each iteration of the stochastic gradient descent algorithm [55, 39].

For this test, the Square and Crop modes have been both analyzed, declaring strategic dimensions for the data input to the network. In particular, the training predictions have been compared in Square@1024 and Square@1792 modes with those in Crop@512 and Crop@1024 mode, where the suffix indicates the dimensions of the image side in input to the network. The size of the input image side has been chosen by comparing the maximum value of the image side (i.e., 1792 for Square mode) with lower values, multiples of 64, that are used most frequently for the convolutional neural networks training in the computer vision field. The test has been carried out by training the models for at least 150 epochs with a training set consisting of 50 manually annotated images. The final training epoch has been controlled by a customized callback that extended the traditional early stopping [55, 39] regularization tecnique through a moving average on the validation loss. Model

	Detection			Segmentation		
Configuration	mAP	mAR	mAF	mAP	mAR	mAF
Square@640	53.7%	24.7%	33.7%	55.9%	26.8%	36.1%
Pad64	57.3%	40.3%	47.1%	58.8%	41.9%	48.8%

Table 5.3: Effects of different image resolutions on final performances.

inferences have been evaluated on a test set of 110 manually annotated images. Being originally the entire dataset consisting of 550 images (including 410 hand labeled and 140 not labeled), a single test set of 110 images has been maintained, regardless of the number of images used in the different subsets of the training set used to make all the results of the various tests comparable to each other (Section 4.5).

For the inferences, as previously, with respect to the Matterport's implementation [1] of Mask R-CNN [30], it is possible to exploit two types of configuration, Square and Pad64. Square mode in inference works exactly like training and does not need further details. The Pad64 mode, on the other hand, evaluates the images in their original dimensions without any resize with the exception of a zero padding as long as the final dimensions are not multiple of 64. Square mode is used to evaluate the predictions of the models trained respectively in Square@1024 and Square@1792. In this way the images in input to the networks are placed in the same conditions for training and for inference. Pad64 mode, instead, is particularly recommended if the training is done in Crop mode because the original dimensions of the image are maintained without any resize. Among all the various configurations we can see that the Crop@1024 configuration is a good compromise between training times and quality of final predictions. This allows obtaining comparable results in a relatively short time and is a good starting point for experimental tests. All subsequent tests presented in this section refer to this last training configuration.

Another study has evaluated the impact of low vs high resolution images on the final predictions of the model. In particular, although the training set is composed entirely of high-resolution images, the Customer also owns many low-resolution images in its systems. This is due to the fact that in the past the data collection tool in SAP Customer systems involved the acquisition of low- rather than high-resolution images. In order to analyze the entire time series of data for marketing purposes, it is therefore necessary to quantify the effects low-resolution images on the final performance. To provide a comparison metric, the images in the high-resolution dataset mainly are 1792 × 1536, low-resolution images are about 640 × 480.

The study is structured as follows.

- only the high-resolution dataset during training and inference is considered.
- the training is done in Crop@1024 mode as indicated by previous results.
- with the same training model, the performances between the inference in Pad64 mode and in Square@640 mode are compared.

This takes advantage of the scaling provided by the Square@640 mode to see how low-resolution images impact the final results. In particular, it is clear that, keeping the model configurations unaltered, all the performances degrade if the input images have low resolution. As we can observe in Table 5.3, the main critical point is the recall, which decreases considerably. This also strongly depends on the other configuration parameters of the model, and probably also on the fact that the input dataset consists mainly of high resolution images. Further tests are currently planned in order to investigate this problem more accurately and correct it.

5.2.2 Dataset Studies

As discussed in Section 4.5, the creation of the dataset is a central problem in supervised learning and in particular in the training of modern neural networks. The main experimental results on

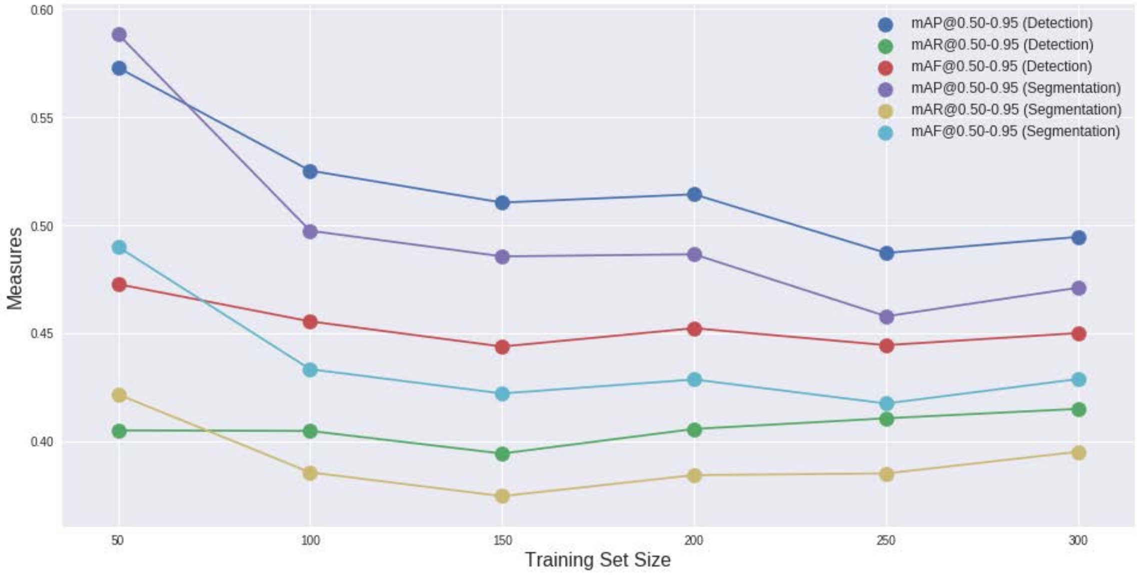


Figure 5.1: Effects of automatically annotated images on network performances.

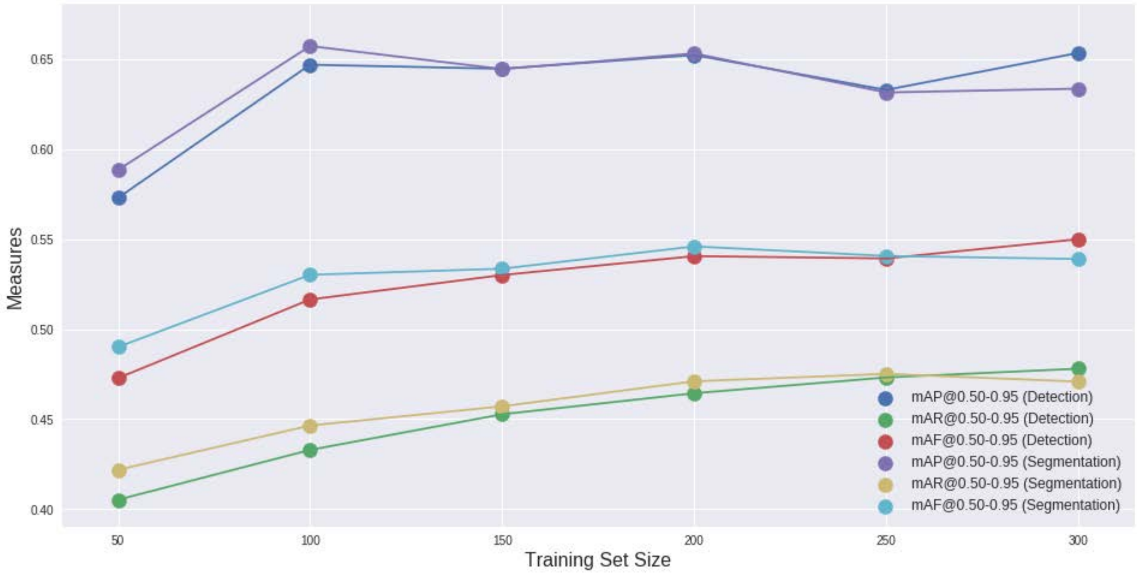


Figure 5.2: Effects of manually annotated images on network performances.

the construction of the dataset are presented below. The experiments involved the neural network trained on images of generically labeled products, that is, by maintaining a single label rather than discriminating for each class. This made it possible to obtain a more accurate evaluation of the results for small training sets. The results have been subsequently scaled in the multi-class context and will be presented in the following sections. All evaluations refer to a single test set consisting of 110 manually annotated images.

In terms of automation of the dataset construction process the first experiment wants to investigate the effects of images automatically annotated by the network on the final performances. The automatic annotations have been obtained by running the network inferences on the unlabeled images and obtaining the segmentations and the bounding boxes of the various products directly from the latter through image processing techniques. Starting from a baseline of manually an-

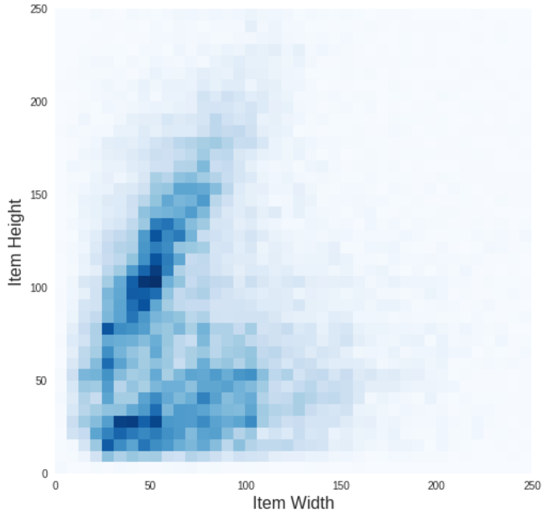


Figure 5.3: Ad-Hoc dataset insights on different items scales.

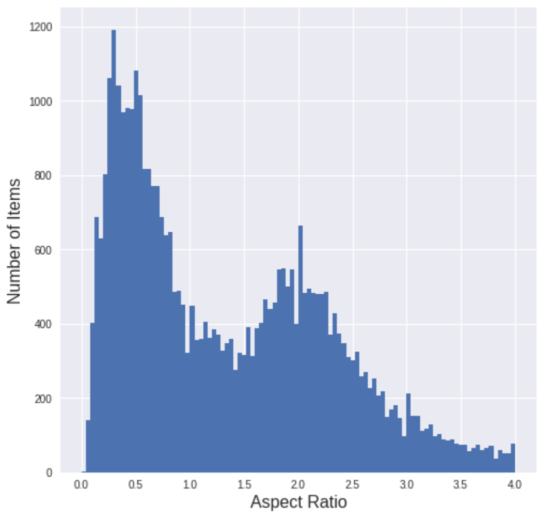


Figure 5.4: Ad-Hoc dataset insights on different items aspect ratios.

notated images, the graph presented in Figure 5.1 presents the mAP, mAR, mAF measurements averaged on the IoU range [0.5 : 0.95] over subsequent batches of automatically annotated images. In particular, starting from 50 manually annotated images, the performances on the metrics have been monitored by adding batches of 50 images automatically labeled by the network to reach a total of 300 training images. All the models have been also evaluated through the Wilcoxon signed rank test compared to the base model trained on the 50 manually annotated images in order to ensure that predictions do not come from the same distribution. As it can be seen, the performances obtained from this experimental test are disappointing for both evaluations carried out on the bounding box and segmentation tasks. In particular, mAP and mAF decrease each time a further batch of automatically annotated images is added. The automatic annotation in the Customer use-case does not therefore constitute a valid approach to decrease the number of necessary annotations. Further tests are planned in these scenarios to investigate the effects of data augmentation techniques and the use of synthetic datasets on the final performances.

In order to determine quantitative information on the number of images necessary to obtain satisfactory performances in the Customer use-case, the previous test has been replicated with fully hand-annotated images. In particular, for this test the same images (albeit not the same annotations) have been used, and the same batches of the previous test have been kept. It can be observed in Figure 5.2 that in this case, the addition of further images to the training set tends to improve all the metrics of interest. As expected, this improvement ends to saturate as more images are added to the training set. From this test, it emerges that currently for the Customer use-case, the addition of manually annotated images is the only method that can improve inference performance by increasing the size of the dataset.

5.2.3 Fine-Tuning

Fine-tuning is a crucial method for improving model inference performance (Subsection 2.3.3). In particular, experimental tests may be performed to determine the optimal configuration of the network architecture on a predefined search space. It is possible to use complex techniques based on grid search or random search [39] but in our case, given the very long training times, we decided to investigate a parameter space defined ad-hoc. The following tests further extend the survey dimensions provided by the ablation experiments of the network architectures presented in the previous sections. All tests have been performed on a reduced dataset of 50 images to optimize the training time while maintaining comparable performance. These refer to the same test set

	Detection			Segmentation		
Anchor Scales	mAP	mAR	mAF	mAP	mAR	mAF
{2, 4, 8, 16, 32}	53.5%	30.9%	39.0%	54.8%	32.2%	40.5%
{4, 8, 16, 32, 64}	53.4%	32.8%	40.5%	53.6%	33.7%	41.3%
{8, 16, 32, 64, 128}	57.3%	40.3%	47.1%	58.8%	41.9%	48.8%
{16, 32, 64, 128, 256}	**62.9%**	**42.6%**	**50.6%**	**62.2%**	**42.7%**	**50.5%**
{32, 64, 128, 256, 512}	64.2%	41.4%	50.2%	64%	41.3%	50.1%

Table 5.4: Instance segmentation results using different anchor scales settings.

	Detection			Segmentation		
Anchor Ratios	mAP	mAR	mAF	mAP	mAR	mAF
{0.3, 1, 2}	58.7%	40.8%	47.4%	60.0%	41.2%	48.7%
{0.3, 1, 2.5}	55.5%	36.8%	44.1%	57.1%	38.4%	45.9%
{0.5, 1, 2}	57.3%	40.3%	47.1%	58.8%	41.9%	48.8%
{0.5, 1, 2.5}	**60.1%**	**40.4%**	**48.2%**	**61.7%**	**42.1%**	**50.0%**

Table 5.5: Instance segmentation results using different anchor ratios settings.

consisting of 110 manually annotated images presented in the previous sections. Also in this case dataset annotations maintain a single label rather than discriminating for each class to obtain more accurate results for small training sets and the mAP, mAR and mAF measurements are averaged on a prefixed IoU range [0.5 : 0.95]. For all the tests presented below, precision/recall trade-off has been investigated in detail as the main limit identified in the architecture. In particular, it is evident that while for any configuration the precision is relatively satisfactory, recall is the main problem. This may be due to various issues and below we will present the main investigation areas.

Mask R-CNN [30] has many parameters to be configured appropriately. Two of these are the dimensions of anchor scales and anchor ratios of the output RPN [59] regions. For this test, a basic configuration has been first defined by visual inspection of the histograms in Figures 5.3 and 5.4. These show the dimensions of the survey parameters of the candidate regions in the ad-hoc dataset. As we can observe, the data are sufficiently concentrated to suggest a basic configuration for anchor scales and anchor ratios depending on the two distributions analyzed. Starting from the basic configuration, the parameter space has been modified to check the performance of adjacent configurations. All configurations have also been chosen according to the constraints of the feature pyramid network based overall topology that requires only 5 values for the anchor scales. Furthermore, another constraint has been imposed by the fine-tuning of the region proposals network on top of the FPN [44] which, previously trained on the Ms COCO [46], required only 3 values for the anchor ratios. Tables 5.4 and 5.5 show the performances of the various configurations tested and the configurations that show the best precision/recall trade-off are highlighted. In particular, as expected, the results tend to prefer the configurations that capture the maximum variance between the data – anchor scales:{16, 32, 64, 128, 256}, anchor ratios:{0.5, 1, 2.5}. These values will be used in the final configuration presented later.

The precision/recall trade-off has been investigated in detail with subsequent tests. For these tests, the basic configuration starts from the optimal values determined in the ablation experiments presented in the previous sections. Subsequently, the search space has been investigated in more detail, trying to remedy the main limit identified in the system. In particular, given the architecture of Mask R-CNN and more generally of modern detectors, in the Customer context is particularly difficult for the network to detect occluded objects in the background. This is mainly due to the overlapping limit of regions dictated by the NMS thresholds of the RPN and the final layers, respectively. Other related search parameters result to be the number of region proposals in output to the RPN and the confidence for which the final inferences can be considered valid. All these

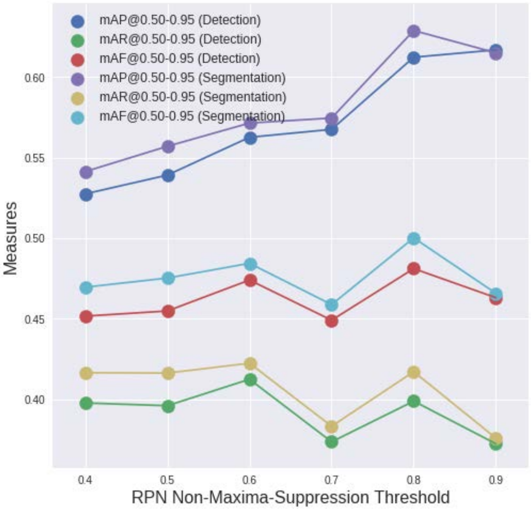


Figure 5.5: Instance segmentation results using different RPN non-maxima suppression thresholds.

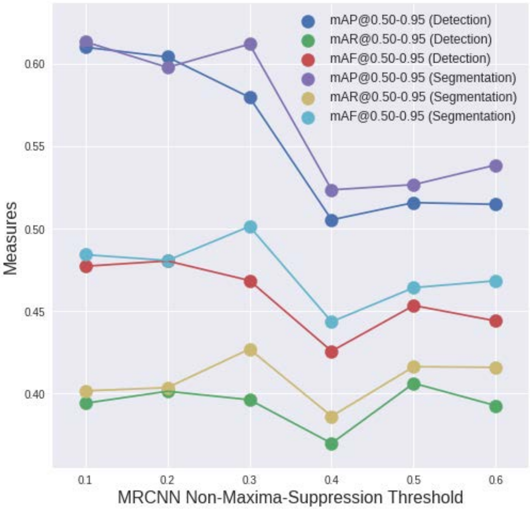


Figure 5.6: Instance segmentation results using different MRCNN non-maxima suppression thresholds.

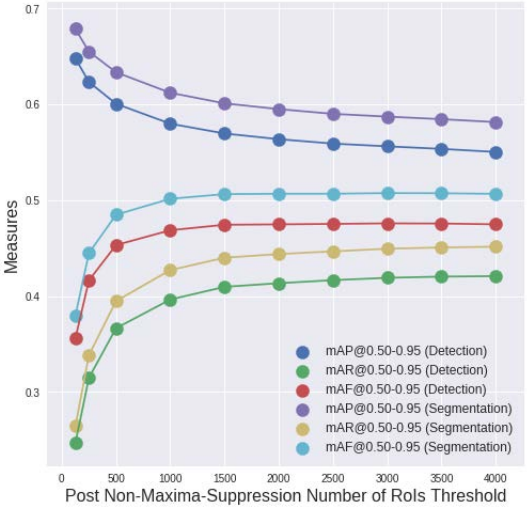


Figure 5.7: Instance segmentation results using different post non-maxima suppression number of RoIs threshold.

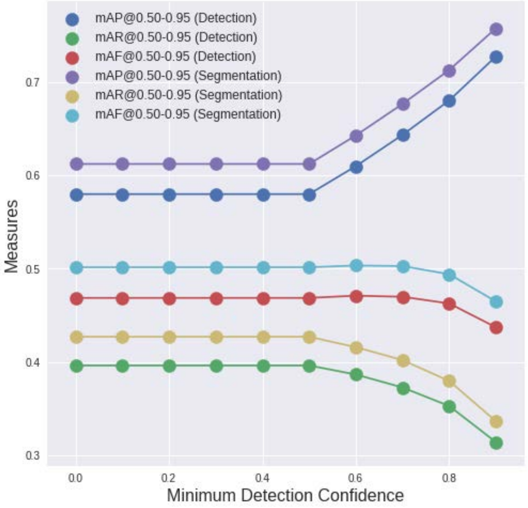


Figure 5.8: Instance segmentation results using different minimum detection confidences.

tests are presented in Figures 5.5, 5.6, 5.7 and 5.8. As can be observed for the NMS, in both cases (Figures 5.5 and 5.6) the precision tends to increase as the NMS threshold is higher. The latter consideration should be contextualized keeping in mind that in the Matterport's implementation [1] of Mask R-CNN [30], the NMS threshold applied subsequently to the RPN layers is higher as much as the value is close to 1 while the same threshold applied subsequently to the final layers (MRCNN) is higher as much as the value is close to 0. In both cases, this does not seem to affect the recall too heavily and it is possible to find a compromise that maximizes the F-Measure by imposing as thresholds 0.8 and 0.2, respectively. Concerning the number of RoIs maintained after the NMS phase in output to the RPN and the minimum detection confidence required so that an instance in output to MRCNN can be considered positive (Figures 5.7 and 5.8), it is also possible to define two thresholds. In particular, as previously, the values that maximized the F-Measure have been selected, preferring a greater recall in case of multiple candidate values. This has been solved

	Gnrc	A	B	C	D	E	F	G	H	Mean
mAP@0.5:0.95	55.9%	80.9%	82.7%	70.8%	73.8%	70.4%	75.6%	68.9%	76.6%	72.8%
mAR@0.5:0.95	42.9%	75.3%	75.0%	68.8%	69.3%	61.5%	62.8%	57.0%	65.6%	64.3%
mAF@0.5:0.95	48.4%	78.0%	78.6%	69.7%	71.4%	65.4%	68.4%	62.4%	70.5%	68.1%

Table 5.6: Per-class detection results on multiple class scenario.

	Gnrc	A	B	C	D	E	F	G	H	Mean
mAP@0.5:0.95	57.0%	81.1%	81.2%	72.2%	76.5%	70.5%	76.2%	67.7%	76.5%	73.2%
mAR@0.5:0.95	44.2%	75.5%	75.3%	70.0%	72.3%	64.4%	64.1%	55.4%	68.3%	65.6%
mAF@0.5:0.95	49.6%	78.2%	78.1%	71.0%	74.6%	67.2%	69.5%	60.9%	72.0%	69.0%

Table 5.7: Per-class segmentation results on multiple class scenario.

	Gnrc	A	B	C	D	E	F	G	H	Mean
mAP@0.5:0.95	48.7%	82.4%	79.2%	72.8%	5.65%	8.12%	77.4%	22.2%	71.6%	52.0%
mAR@0.5:0.95	39.6%	52.0%	45.5%	17.5%	48.2%	44.1%	8.55%	33.6%	14.5%	33.7%
mAF@0.5:0.95	43.5%	63.8%	57.7%	28.2%	10.1%	13.7%	15.4%	26.7%	24.2%	40.9%

Table 5.8: Per-class aggregated detection results on multiple brand scenario.

	Gnrc	A	B	C	D	E	F	G	H	Mean
mAP@0.5:0.95	49.7%	82.4%	78.1%	79.1%	5.80%	8.16%	79.2%	21.9%	66.2%	52.3%
mAR@0.5:0.95	40.8%	51.1%	45.3%	19.2%	50.4%	45.6%	8.61%	33.3%	13.4%	34.2%
mAF@0.5:0.95	44.7%	63.1%	57.3%	30.8%	10.4%	13.8%	15.6%	26.4%	22.3%	41.4%

Table 5.9: Per-class aggregated segmentation results on multiple brand scenario.

by choosing 1500 as the number of RoIs limit in output to the RPN and 0.5 as the threshold for the minimum detection confidence for which an instance in output to MRCNN can be considered positive. As previously, all these values will be used for the final configuration tests presented later.

The tests discussed above have investigated several configurable parameters in the network architecture, but further tests are planned to extend the survey area.

5.2.4 Final Configuration

The previous results gave indications for the the main configuration parameters of the network architecture. The final results in the context of multi-class inference are shown below by taking into consideration the best model configurations previously identified. It is important to note that in the dataset (Figure 4.7) superclasses of Customer products have been defined that are sufficiently large to obtain, in most cases, significant results in output. The list of products has been previously agreed directly with the Customer and represent the core business of the Customer. The results of the Mask R-CNN network inferences trained in a multi-class scenario have been compared with those of the Mask R-CNN network trained in a single-class case but aggregating the last inferences through the double stage architecture presented in the previous sections. For the test, in both inference scenarios, performance by class and aggregate final performance are reported. A further test was also planned to evaluate the effectiveness of our detector based on Mask R-CNN compared to that presented by Tonioni [71] but it was not possible to perform it because the dataset mentioned in the paper turned out to be proprietary and could not be disclosed to us in compliance with the privacy policies.

In Tables 5.6 and 5.7 we can examine the per-class instance segmentation results as well as the final ones obtained by evaluating the inferences of the network composed only by the Mask R-CNN [30] stack. In particular, it is evident that the performance of the network in the recognition of Customer products is much higher than that on generic competitors products. These results are very important because they can suggest a conceptual link to the occlusion and illumination issues mentioned in Chapter 3. At present, no formal metric has yet been defined for the Ad-Hoc dataset to measure it but examining in a visual way the dataset we can indeed notice that Customer products are always positioned in the foreground and there are no annotations for the Customer products in background in the test set. In particular, by examining the products in the background we can see that the annotations in the dataset belong exclusively to the competitors product class (Gnrc) and the corresponding instances are often subject to occlusion phenomena as well as to low light conditions. Overall, even if recall is still the most critical aspect, the results obtained are satisfactory for all classes and constitute a good solution for the analysis of the Customer brand that we will present in the following section.

In order to evaluate the quality of the brand detector with respect to the product detector, it has been decided to aggregate the brand detector inferences on the same superclasses of products identified by the product detector. This decision has been adopted to make the results of the two detectors comparable. In particular, the main constraint that motivates this choice is due to the fact that currently the Ad-Hoc dataset does not have a number of annotations adequate to be able to compare the two detectors inferences on the different subclasses since only the brand detector could achieve good results in this challenge. In addition, given that products have great variability among themselves within the same superclass according to the type of package or promotional campaign, a class balancing attempt has been made before registering the final inferences of the brand detector. In particular, the query dataset used in the classification pipeline of the brand detector has been built in a balanced way by inserting 10 different instances for each superclass of Customer product and 50 different instances for competitors products under various conditions. This change in the construction of the query dataset that deviates from the work initially proposed by A. Tonioni [70, 71], has been carried out after recording very disappointing results (not shown) if a single product image was maintained for each subclass in the dataset. In Tables 5.8 and 5.9 we can see the final results. It is apparent that the performances are not yet comparable with those of the product detector – Tables 5.6 and 5.7 – nor with those registered by Tonioni [70, 71]. However, the approach offers potential for improvement if developed more carefully. In particular, it is possible, for example, to fine-tune the network architecture for feature extraction, or to use specialized heuristics. Moreover, one of the fundamental observations that could be made refers to the fact that currently, there are no good quality product images in the query dataset. This last fact is due mainly to the impossibility of currently having, for lack of authorization, good quality images from the Customer's products catalog but soon the problem will be studied in greater detail as described in Chapter 6. In any case, the brand detector approach actually remains the only architecture still available to recognize the product brand without exploiting expensive amounts of annotations in the Customer business domain.

5.3 Business Intelligence and Visualization

Business Intelligence (BI) includes all the business processes that aim to collect data and analyze strategic enterprise information. BI technologies are used for different purposes, such as as data analytics, performance benchmarking, process mining and predictive analytics. The results are traditionally made available in the form of automated reporting and allow to support a wide range of business decisions. These include product positioning, pricing, priorities, targets and all the key insights that allow a company to gain a competitive market advantage over its competitors.

In our domain, business intelligence results are available in respect of the privacy of Customer and Supplier data and therefore sensitive information is obscured. The results are currently made available in cloud through SAP Analytics Cloud for direct consultation (Section 4.4). Data are presented in the form of storytelling and the main pages have been briefly described in Section 4.3, dedicated to the presentation logic. With reference to the mock-ups, in addition to the initially agreed pages, others have been implemented. For brevity, the various pages are not presented individually, but the main results available are described in detail. In order to analyze a single business process, the following results refer to an historical series initially defined composed by

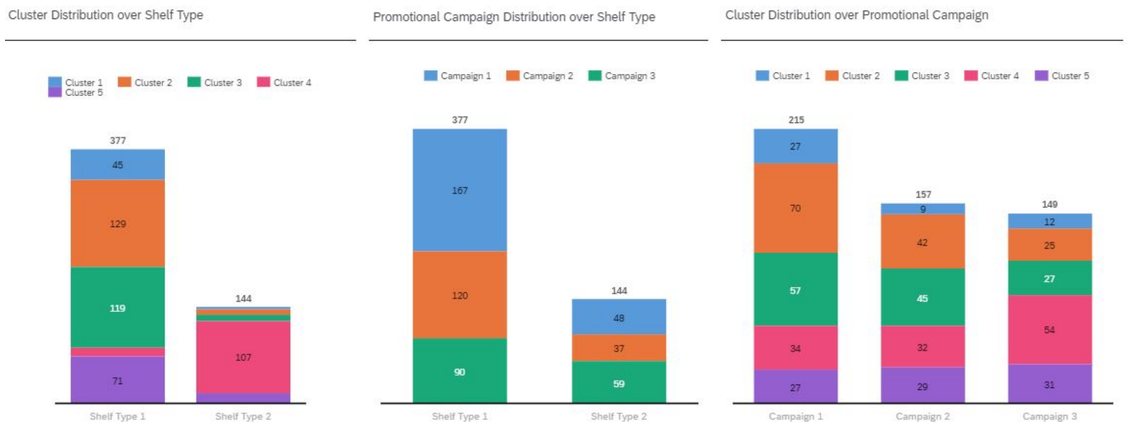


Figure 5.9: Customer basic KPIs.

521 Customer activities. This series has been chosen to make the results available in the form of a Proof Of Concept mainly composed of high resolution images. These results will be integrated with the analysis of other business processes as soon as the Customer makes the data available. Initially, these results will be extended to the historical series of all the activities available in the IT Customer systems. Subsequently, the data will have a broader investigation area and it will be possible to correlate the single product present within an activity with the revenue of each pharmacy. The results presented below are aimed at enhancing the potential of AI as a tool for analyzing business processes. Basic KPIs have been defined which can be obtained by viewing the questionnaires associated with the activities. More advanced KPIs have been obtained only through AI image processing and deep learning techniques.

With reference to the requirements defined in Section 4.1, all functional and non-functional requirements have been implemented with respect to the agreed constraints. Regarding specifically the visual analytics requirements about KPIs verification matching on the deep learning elaborations results with the questionnaires it has been decided to not include the information in this work at the moment. The requirement has been correctly implemented as requested but since the data entered in the questionnaires by commercial agents were often incorrect or incomplete, it was not possible to obtain reliable measurements. However, a specific activity has been planned to overcome this lack of information which involves re-elaborating the whole series of questionnaires considered for the considered Customer's business processes.

5.3.1 Basic KPIs

Results described in Figure 5.9 refer to the information previously available to the Customer. The information contained in the questionnaires attached to the activities were aggregated and presented in the form of histograms to define strategic metrics of interest. In particular, final results allow to observe the distribution of product clusters and promotional campaigns over the shelf type. This information is very important because it is the basis of further recommendation processes on specific market targets. However, although a standard document was available to regulate the type of information to be included in the Customer systems, the agents in charge were not sufficiently precise and the data entered were occasionally inaccurate or incorrect. In this case AI is a useful tool for more objective and accurate analysis, being able to provide the same information by applying computer vision techniques to the images associated with the activities.

5.3.2 Advanced KPIs

More advanced KPIs and AI-based metrics provide a greater level of detail than previous ones. In an era where information is considered a very important asset, companies invest large sums of

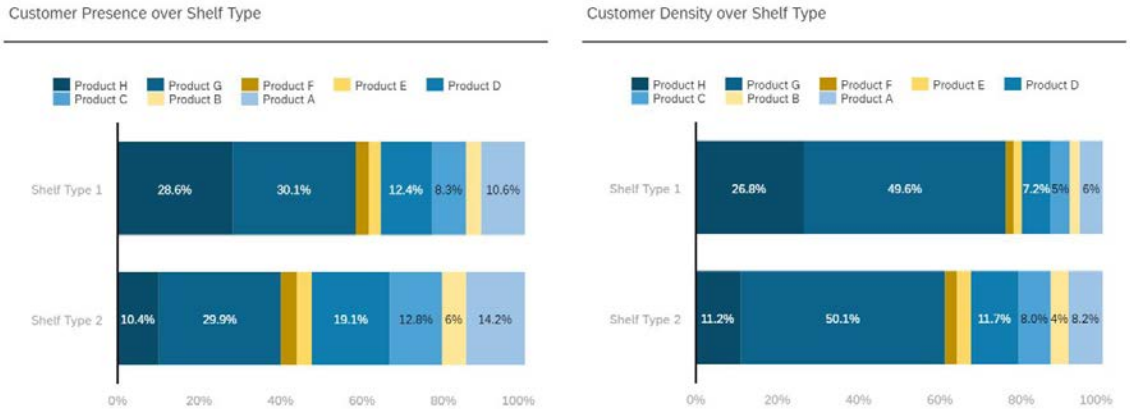


Figure 5.10: Per product Customer brand influence over shelf type.

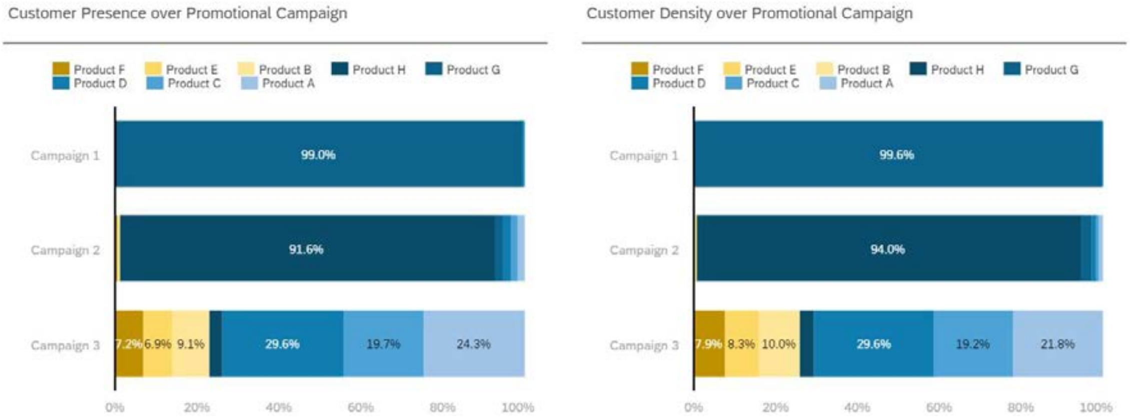


Figure 5.11: Per product Customer brand influence over promotional campaign.

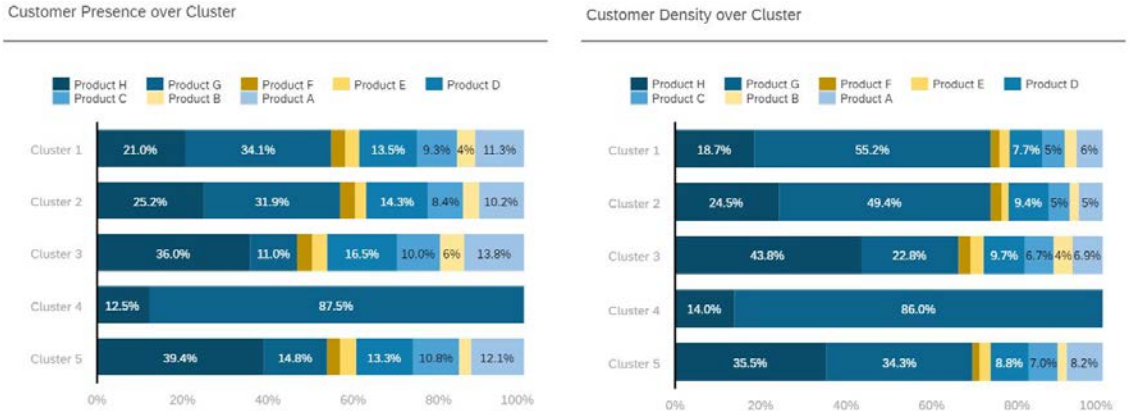


Figure 5.12: Per product Customer brand influence over cluster.

money to obtain strategic information that can guarantee a competitive advantage in the market. Through this information it is possible to promptly move large flows of operations and capital to obtain business results quickly. The designed system arises the objective of obtaining intelligence information useful for making strategic decisions in the Customer's business processes. Some of the currently available key insights will be presented below.

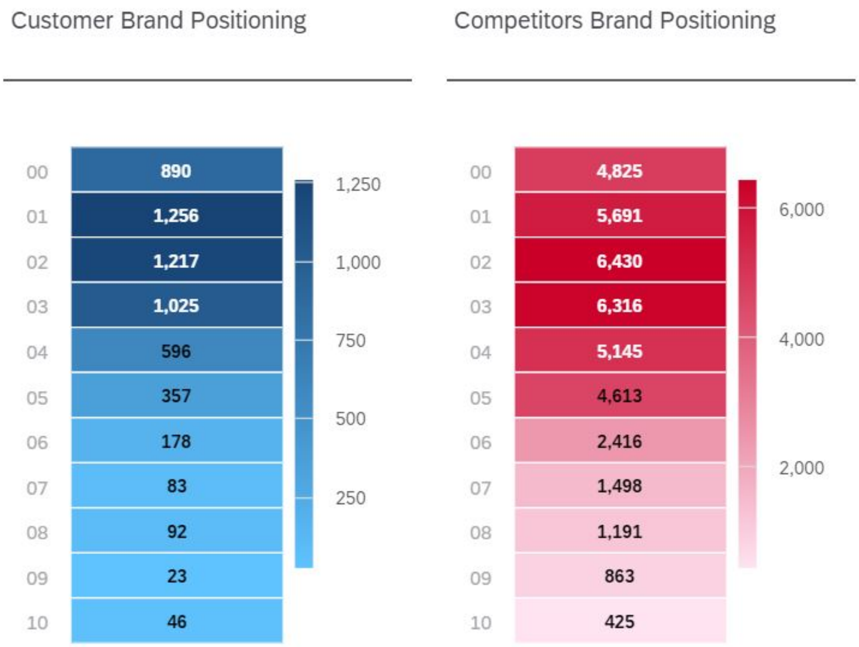


Figure 5.13: Products insights on store shelves.

The diagrams in Figures 5.10, 5.11 and 5.12 analyze the presence of Customer products in pharmacies in terms of number of identified objects and number of pixels. This allows to obtain a whole series of key insights on the Customer's activity, suitably aggregating the inferences of the algorithm. In particular, the products identified in the images are aggregated for shelf type, promotional campaign and cluster. It must be noted that the importance of the brand calculated by number of products is not always the same as the one calculated by number of pixels per product. This is particularly evident if we take into consideration product G which assumes strategic importance in the Customer brand together with product H. In fact, product G and product H categories make up the majority of the area exposed by the Customer in pharmacies regardless of the shelf type and the considered cluster. Even cluster 4 is entirely dedicated to product G and product H and does not host other product categories. Moreover, specific promotional campaigns aim to promote these two products compared to the rest, as we can see in Figure 5.11. However, all the remaining products constitute, if considered as a whole, a very influential range of products in the Customer brand. These are present at the same time, each with different importance, in the final pharmacies and also to products A, B, C, D, E, F (considered together) is dedicated a promotional campaign targeted on a specific market segment. Other types of insights can be obtained by correlating these KPIs with the knowledge of the Customer's business domain. In particular, it is very important to note that in the aggregated results for promotional campaigns, campaign 1 and campaign 2 are not exclusively dedicated to product G and product H. Furthermore, there is a significant presence of product H also in campaign 3. From cross-analysis directly on the images we can see that the outliers identified by the algorithm are correct. This is particularly problematic for the Customer brand and indicates that the agents appointed to set up the stores performed mistakes and even harmful operations. On the basis of this information it is possible to define tolerance thresholds and promptly intervene in the single pharmacy through maintenance operations to resolve any dangerous situations that may arise. Furthermore, by correlating these KPIs with the revenues of the individual sales points, it will be possible to provide future recommendations for the preparation of set-ups in pharmacies.

Other metrics of strategic importance for the Customer concern the products positioning on each store shelf. As with the previous case it is possible to extract this information directly from the images through the designed system. Results are always available via automated reporting

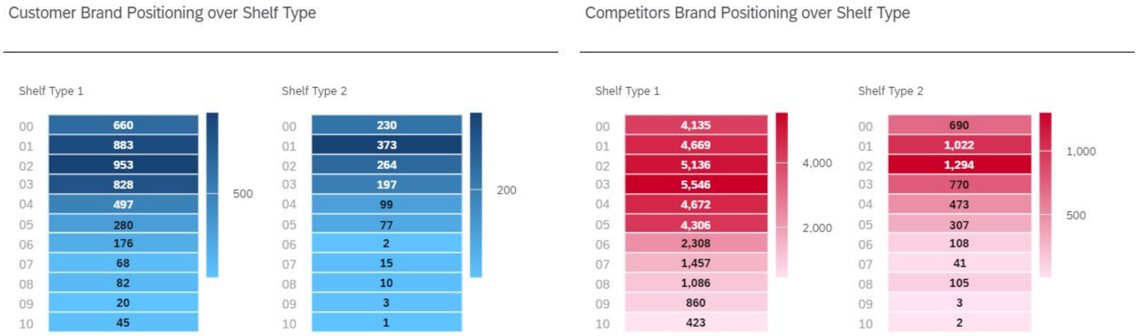


Figure 5.14: Products insights on store shelves over shelf type.

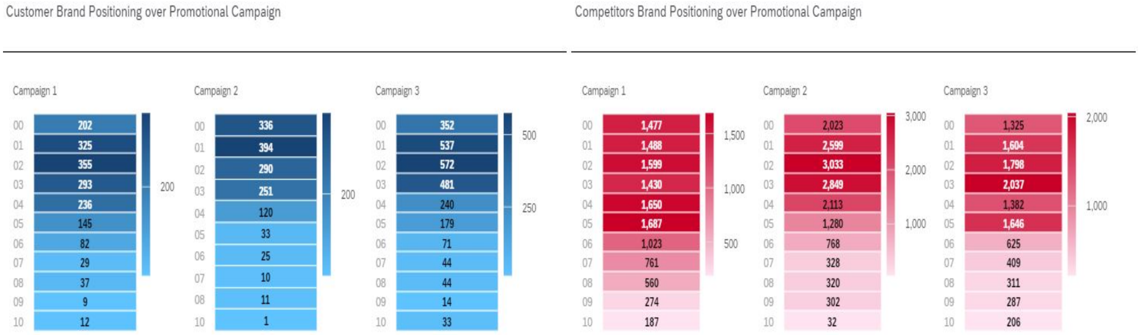


Figure 5.15: Products insights on store shelves over promotional campaign.

by aggregating data according to the desired level of detail, as shown in Figures 5.13, 5.14 and 5.15. In particular, as we can see in Figure 5.13, it is immediately important to note that while Customer products tend to be concentrated in the upper shelves, those of the competitors cover a larger surface area. This insight provides a competitive advantage because traditionally products positioned at face height have greater visibility. Other key insights can be obtained from the analysis of products positioning on store shelves over shelf type and promotional campaign. In particular, it can be noted that both for the Customer and for the competitors, shelf type 2 tends to concentrate the products in the highest shelves while shelf type 1 tends to disperse them over a larger surface (Figure 5.14). The dispersion of the products also depends on the promotional campaign considered, as it is possible to observe from Figure 5.15. In particular, the Customer products identified in campaign 2 are positioned too high with respect to those of the competitors. This is particularly problematic because products positioned too high are less visible and this has a negative effect on final sales and on the Customer brand. Similar information had already been correlated in the past with the sales potential of the Customer products and in this case there is definitely a positive feedback.

The potential of the designed system is greater compared to the system previously available for Customer where only the basic KPI information is present. Starting from these considerations many other key insights based on the designed system will be available in the future. In particular, one of the main objectives is to determine the ideal configuration for each product and for each shelf type and promotional campaign based on sales forecasts. With respect to this last point, all the considerations on future developments of this work will be discussed in Chapter 6.

Chapter 6

Conclusion

This work proposes an automated retail shelf analytics system based on modern deep learning and image processing techniques (Chapters 4 and 5). In particular, the system model is based on Mask R-CNN [30] and allows to obtain instance segmentation inferences of shelf products from store pictures. The network architecture has been customized through fine tuning to adapt to the agreed business domain and the system can analyze business processes images at scale detecting and segmenting each product instance. Starting from the image processing, business intelligence KPIs are calculated to provide key insights on the retail business world (Section 5.3). The system has been designed in collaboration with the University of Padua for a Customer company that plays a role of strategic importance in the international pharmaceutical retail market. The work has been sold from a Supplier consulting company according to a business integration project between the IT Customer and Supplier systems. The system consists of an algorithmic kernel based on TensorFlow [51, 1] and integrates directly into a SAP architecture to manage both server and client sides data flows from the two business systems. Starting from this work many developments will follow in the coming months to further improve the promotional activities that Customer performs daily.

A part of this work that has been a precious time investment and that contributed to the success of this project has been the creation of a proprietary instance segmentation dataset (Section 4.5), built ad hoc to effectively adapt the system model inferences to the Customer business domain. In addition to manual pixel-wise labeling of all images in the dataset, various design alternatives have been examined in detail, focusing on the trade-off between the accuracy of predictions and the amount of annotations required for the training of the neural networks. With respect to this point have been studied, the effects of the automatic labeling of the dataset images on final predictions (Subsection 4.4) that can be integrated into the Mask R-CNN chain of inferences able to identify products, on store shelves without the need to own large amounts of data for the neural networks stack training. Both these latest design alternatives will be studied in more detail in future developments to obtain more specialized results on the business domain.

Another part that proved to be very important for this work and which has been a strategic investment both in terms of time and money has been the design of a business integration system (Section 4.4) between the IT Customer and Supplier systems. In particular, besides having followed all the design phases of an R&D solution that integrates the traditional Customer's business processes, also all the consulting activities with the various Customer departments have been taken care of. According to specifications, the design has been conceived in a full-stack perspective and is based on a data flow between the SAP systems, which is aimed at the automated reporting of results in cloud through a dedicated SAP architecture. From the processing, business intelligence KPIs are extracted for the assessment of the impact of the Customer brand in the stores (Section 5.3) according to different interpretations meeting all the functional and non-functional requirements defined in the requirements analytics phase (Section 4.1). In compliance with the confidentiality constraints imposed by Customer and Supplier, some of the most important details with regards to the business integration project have been obscured to defend the intellectual property of the various parties where required.

6.1 Future Work

Future work will see an even more specialized data-driven integration within the Customer's business processes. As a first integration of the current system further KPIs will be defined to evaluate the sales performance of the products on different customer target groups. Based on the defined KPIs and the sales forecasts of the products in the various geographical areas, a recommendation system will be studied to suggest promotions and discounts to the various customers. Following an integration to the system able to provide managerial guidelines for the optimal arrangement of products on store shelves, based as for the previous points on sales forecasts, will be proposed. The system will be designed for the Customer's marketing and sales department and will allow to make strategic decisions to enhance the visibility effects of the Customer's brand over its competitors. Furthermore, a process automation system able to replace the traditional questionnaires filling process that commercial agents visiting the stores fill out daily will be studied. This system, in addition to calculating a good part of the KPIs that are currently automatically determinable by the designed system model, will be able to determine possible out-of-stock by sending appropriate reports to commercial agents in real time. Even the architecture will be improved both from the algorithmic and the system integration point of view. As far as the algorithmic component is concerned, an R&D activity has been planned for the upgrade of the system model chain of inferences. This will primarily concern a more in-depth study of the deep learning pipeline based on the query dataset proposed in Chapter 4. Through the latter it will be possible to make brand predictions in order to classify all the products of the Customer suite as well as all the competitor products. The identification will subsequently be extended to baselines and prices to obtain further intelligence information able to monitor the promotional activities of Customer. As for the system integration component, in addition to the data layer migration in SAP HANA as described in Chapter 4, another architecture integration activity has been planned through another product of the SAP suite. In particular, the potential of the SAP Data Hub will be exploited to orchestrate the data flows coming from the IT Customer and Supplier systems, obtaining benefits as regards the scalability of the whole system and an easier prototyping of new features.

Bibliography

[1] Waleed Abdulla. *Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow.* `https://github.com/matterport/Mask_RCNN`. 2017.

[2] David Acuna et al. "Efficient Interactive Annotation of Segmentation Datasets with Polygon-RNN++". In: (2018).

[3] Amaury. *Pixel Annotation Tool.* `https://github.com/abreheret/PixelAnnotationTool`. 2017.

[4] Soheil Bahrampour et al. "Comparative study of deep learning software frameworks". In: *arXiv preprint arXiv:1511.06435* (2015).

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features". In: *European conference on computer vision.* Springer. 2006, pp. 404–417.

[6] Michael Buhrmester, Tracy Kwang, and Samuel D Gosling. "Amazon's Mechanical Turk: A new source of inexpensive, yet high-quality, data?" In: *Perspectives on psychological science* 6.1 (2011), pp. 3–5.

[7] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. "An analysis of deep neural network models for practical applications". In: *arXiv preprint arXiv:1605.07678* (2016).

[8] Cyril W Cleverdon. "The significance of the Cranfield tests on index languages". In: *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval.* Citeseer. 1991, pp. 3–12.

[9] Cyril W Cleverdon, Jack Mills, and E Michael Keen. "Factors determining the performance of indexing systems,(Volume 2: Test Results)". In: *Cranfield: College of Aeronautics* (1966), p. 28.

[10] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)". In: *arXiv preprint arXiv:1511.07289* (2015).

[11] *Computer Vision Datasets on the Web.* URL: `http://www.cvpapers.com/datasets.html`.

[12] Thomas M Cover, Peter E Hart, et al. "Nearest neighbor pattern classification". In: *IEEE transactions on information theory* 13.1 (1967), pp. 21–27.

[13] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[14] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *international Conference on computer vision & Pattern Recognition (CVPR'05).* Vol. 1. IEEE Computer Society. 2005, pp. 886–893.

[15] A. Dutta, A. Gupta, and A. Zissermann. *VGG Image Annotator (VIA).* `http://www.robots.ox.ac.uk/~vgg/software/via/`. 2016.

[16] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results.* http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[17] Patrick Follmann et al. "MVTec D2S: Densely Segmented Supermarket Dataset". In: *arXiv preprint arXiv:1804.08292* (2018).

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning.* Vol. 1. 10. Springer series in statistics New York, NY, USA: 2001.

[19] Kunihiko Fukushima and Sei Miyake. "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets.* Springer, 1982, pp. 267–285.

[20] Alberto Garcia-Garcia et al. "A review on deep learning techniques applied to semantic segmentation". In: *arXiv preprint arXiv:1704.06857* (2017).

[21] Marian George and Christian Floerkemeier. "Recognizing products: A per-exemplar multi-label image classification approach". In: *European Conference on Computer Vision.* Springer. 2014, pp. 440–455.

[22] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision.* 2015, pp. 1440–1448.

[23] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2014, pp. 580–587.

[24] Ian J Goodfellow et al. "Maxout networks". In: *arXiv preprint arXiv:1302.4389* (2013).

[25] Google. *Google Trends.* 2012. URL: `http://trends.google.com/trends`.

[26] Dhruv Grewal, Anne L Roggeveen, and Jens Nordfält. "The future of retailing". In: *Journal of Retailing* 93.1 (2017), pp. 1–6.

[27] Donna Harman. "Information retrieval evaluation". In: *Synthesis Lectures on Information Concepts, Retrieval, and Services* 3.2 (2011), pp. 1–119.

[28] John C Hay, Ben E Lynch, and David R Smith. *Mark I perceptron operators' manual.* Tech. rep. CORNELL AERONAUTICAL LAB INC BUFFALO NY, 1960.

[29] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 770–778.

[30] Kaiming He et al. "Mask r-cnn". In: *Computer Vision (ICCV), 2017 IEEE International Conference on.* IEEE. 2017, pp. 2980–2988.

[31] Kaiming He et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition". In: *European conference on computer vision.* Springer. 2014, pp. 346–361.

[32] Jonathan Huang et al. "Speed/accuracy trade-offs for modern convolutional object detectors". In: *IEEE CVPR.* Vol. 4. 2017.

[33] David H Hubel and Torsten N Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". In: *The Journal of physiology* 160.1 (1962), pp. 106–154.

[34] David H Hubel and Torsten N Wiesel. "Receptive fields of single neurones in the cat's striate cortex". In: *The Journal of physiology* 148.3 (1959), pp. 574–591.

[35] Straka Huyen. "Stanford University CS20: Tensorflow for Deep Learning Research". In: (). URL: `http://web.stanford.edu/class/cs20si/syllabus.html`.

[36] John Illingworth and Josef Kittler. "A survey of the Hough transform". In: *Computer vision, graphics, and image processing* 44.1 (1988), pp. 87–116.

[37] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. "Quality management on amazon mechanical turk". In: *Proceedings of the ACM SIGKDD workshop on human computation.* ACM. 2010, pp. 64–67.

[38] Philipp Jund et al. "The Freiburg Groceries Dataset". In: *arXiv preprint arXiv:1611.05799* (2016).

[39] Andrej Karpathy. "Stanford University CS231n: Convolutional Neural Networks for Visual Recognition". In: (). URL: `http://cs231n.stanford.edu/syllabus.html`.

[40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[41] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. IEEE. 2006, pp. 2169–2178.

[42] LeCun. “THE MNIST DATABASE of handwritten digits”. In: *http://yann.lecun.com/exdb/mnist/* (). URL: %5Curl%7Bhttps://ci.nii.ac.jp/naid/10027939599/en/%7D.

[43] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[44] Tsung-Yi Lin et al. “Feature Pyramid Networks for Object Detection.” In: *CVPR*. Vol. 1. 2. 2017, p. 4.

[45] Tsung-Yi Lin et al. “Focal loss for dense object detection”. In: *IEEE transactions on pattern analysis and machine intelligence* (2018).

[46] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[47] Wei Liu et al. “Ssd: Single shot multibox detector”. In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[48] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[49] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[50] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. “Rectifier nonlinearities improve neural network acoustic models”. In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[51] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[52] Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[53] Marvin Minsky and Seymour A Papert. *Perceptrons: An introduction to computational geometry*. MIT press, 2017.

[54] Trisha Mittal, B Laasya, and J Dinesh Babu. “A Logo-Based Approach for Recognising Multiple Products on a Shelf”. In: *Proceedings of SAI Intelligent Systems Conference*. Springer. 2016, pp. 15–22.

[55] Katanforoosh Ng. “Stanford University CS230: Deep Learning”. In: (). URL: http://cs230.stanford.edu/syllabus/.

[56] Joseph Redmon and Ali Farhadi. “YOLO9000: better, faster, stronger”. In: *arXiv preprint* (2017).

[57] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

[58] Joseph Redmon et al. “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[59] Shaoqing Ren et al. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[60] Frank Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6 (1958), p. 386.

[61] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), p. 533.

[62] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: `10.1007/s11263-015-0816-y`.

[63] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge". In: *International Journal of Computer Vision* 115.3 (2015), pp. 211–252.

[64] Shaohuai Shi et al. "Benchmarking state-of-the-art deep learning software tools". In: *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*. IEEE. 2016, pp. 99–104.

[65] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[66] Jost Tobias Springenberg et al. "Striving for simplicity: The all convolutional net". In: *arXiv preprint arXiv:1412.6806* (2014).

[67] Christian Szegedy et al. "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[68] Christian Szegedy et al. "Inception-v4, inception-resnet and the impact of residual connections on learning." In: *AAAI*. Vol. 4. 2017, p. 12.

[69] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[70] Alessio Tonioni and Luigi Di Stefano. "Product recognition in store shelves as a sub-graph isomorphism problem". In: *International Conference on Image Analysis and Processing*. Springer. 2017, pp. 682–693.

[71] Alessio Tonioni, Eugenio Serro, and Luigi di Stefano. "A deep learning pipeline for product recognition on store shelves". In: *CoRR* abs/1810.01733 (2018). arXiv: `1810.01733`. URL: `http://arxiv.org/abs/1810.01733`.

[72] Jasper RR Uijlings et al. "Selective search for object recognition". In: *International journal of computer vision* 104.2 (2013), pp. 154–171.

[73] Holger Ursin. "The temporal lobe substrate of fear and anger. A review of recent stimulation and ablation studies in animals and humans". In: *Acta Psychiatrica Scandinavica* 35.3 (1960), pp. 378–396.

[74] Srikrishna Varadarajan and Muktabh Mayank Srivastava. "Weakly Supervised Object Localization on grocery shelves using simple FCN and Synthetic Dataset". In: *arXiv preprint arXiv:1803.06813* (2018).

[75] Gül Varol and Rıdvan Salih Kuzu. "Toward retail product recognition on grocery shelves". In: *Sixth International Conference on Graphic and Image Processing (ICGIP 2014)*. Vol. 9443. International Society for Optics and Photonics. 2015, p. 944309.

[76] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *null*. IEEE. 2001, p. 511.

[77] Saining Xie et al. "Aggregated residual transformations for deep neural networks". In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE. 2017, pp. 5987–5995.

[78] Jason Yosinski et al. "How transferable are features in deep neural networks?" In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 3320–3328. URL: `http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf`.

[79] Matthew D Zeiler and Rob Fergus. "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

UNIVERSITÀ DEGLI STUDI DI PADOVA

Department of Information Engineering
University of Padua, Italy
A. De Biasio, Prof. C. Fantozzi

Master Thesis:

Retail Shelf Analytics Through Image Processing and Deep Learning

Thesis type and date:

Master Thesis, February 2019

Supervisor:

Carlo Fantozzi

Student:

Name:	Alvise De Biasio
E-mail:	alvise.debiasio@studenti.unipd.it
Student ID:	1156401