

# UNIVERSITÀ DEGLI STUDI DI PADOVA

**Dipartimento di Fisica e Astronomia "Galileo Galilei"**

**Master Degree in Physics of Data**


**Final Dissertation**

# Physics methods for image classification with Deep Neural Networks


**Internship supervisor**

**Luca Malinverno, MSc**

**Thesis supervisor**

**Prof. Marco Zanetti**

**Candidate**

**Gianmarco Pompeo**


**Academic Year 2019/2020**

*Alla mia famiglia,*
*per avermi sostenuto,*
*incoraggiato, accompagnato*

# Contents

# Acknowledgements

I would like to wholeheartedly thank Luca Malinverno, my internship supervisor, for the timeless support and the never-ending patience shown during these months. This thesis would not have been possible without him: be it in the context of a cozy mountain house or inside an actual office, he has always provided me with much needed intuitions and practical hacks to overcome difficulties but also to appreciate their significance, which can be even more valuable and enlightening.

Heartfelt thanks are also addressed to Porini, the company that hosted my internship, as whole, for the trust placed in me and the openness with which they welcomed me as an inexperienced and disoriented trainee. A particular thought goes to Franco and Omar, who have guided my steps with enthusiasm and kindness.

Last but not least, I am extremely grateful to each and every one of my travel companions throughout these years. It has been a delight to share this ride with all of you and stay reassured: I definitely do not intend to stop enjoying it!

*Padova, 2 april 2021*
*Gianmarco Pompeo*

# Introduction

BIG data, artificial intelligence and neural networks are buzzwords that have become more and more popular over the past decade for various applications, both in research and in business. They all pertain to the digital revolution that has been taking place in the past years: devices and human technology have become increasingly interconnected and they are now capable of generating unprecedented quantities of data (hence "big data"). In turn, this volume of information made it necessary to develop new algorithms which could employ automated learning processes to extract relevant patterns that would otherwise remain hidden; this is the power of artificial intelligence.
Different intelligent algorithms have been designed for a variety of diverse tasks, ranging from object detection to speech recognition, from classification to text analytics, and many more. They do so with little or no human intervention and they are actually so powerful that they often surpass human performances even in the most innate applications. To develop these kinds of algorithms, however, highly-skilled developers and mathematicians are fundamental.

Nevertheless, these new tools have impoverished the analysis process: they act as "black boxes" where data is input and predictions are output, with an excellent degree of reliability. In essence, artificial intelligence algorithms are now so powerful and efficient that have progressively led their users into neglecting to fully understand the single steps that bring the algorithms themselves to successful completion. Such an approach inherently and almost deliberately neglects a deep understanding of the data itself, which however still remains the descriptive building block to characterize whatever is the object of research or development.
A physicist's mindset is to follow a *scientific approach* which requires a meticulous planning of the steps to be followed, a formal definition of descriptive parameters, a quantitative and rigorous analysis to characterize them, a thorough search for robust and clearly-defined processing techniques and, eventually, their application in a controllable and well-understood context. This is the way to follow both when working with typically-sized datasets and with big data.

In order to retrieve the *physics of data*, the goal of the present thesis is to closely follow this scientific approach; the aim is to develop a a novel algorithm to perform detection and classification of goods and products found in supermarkets or grocery shops. The analysis will start out by considering an exemplified toy model with reduced complexity, displaying mono-product images on a white background. This will allow to focus on the understanding of the data structure itself, the information they carry along and how to use it in the most profitable way possible. Images will be treated and transformed employing techniques borrowed from experimental physics in order to extract their relevant features using an innovative approach; afterwards, a custom-built classification procedure will be laid out taking advantage of the observations from the preliminary stage.

More advanced models and solutions will be needed when coping with more generalized and complex scenarios, such as images from supermarket aisles which tend to be defined by a particularly uncontrolled scenario. In this part of the study, artificial intelligence and algorithms based on it will be a key solution, but the line of thinking will remain unchanged. In fact, while it may be hard to have a full understanding of how such complex networks operate, the methodology will still be to have a well-rounded grasp of the variables involved and how

they affect the detection or classification capabilities; this will allow to perform quantitative, evidence-based decisions in selecting the best model to employ and in conditioning it for the specific use case under scrutiny.

This thesis finds a natural environment to see its light in *Porini Srl*, a dynamic business versed in digital consulting and software development which also boasts a partnership with Microsoft, thanks to which a comprehensive infrastructure with several tools and powerful cloud resources is made available.

In this context, research and business tend to blend together reaching an optimal degree of interpenetration. The systematic investigation and the study of materials and sources in order to establish an analysis backbone is always accompanied with a concrete perspective into the business applications that each step is bound to have. Not secondarily, the possibility to exploit state-of-the-art resources and systems indeed grants an overhead also from the computational standpoint, which is brought to a further level of sophistication and which grants competitive benchmarks; the model developed in the first part of the analysis will in fact be compared with Custom Vision AI, a Microsoft image recognition service.

The end result of this collaboration, aside from the results in the present thesis, culminated in the realization of an application to be deployed on mobile devices or computers and containing the model developed for product classification in the controlled toy model. This implementation works as a proof of concept (PoC), meaning a realization of a certain analysis method in order to demonstrate its feasibility and test if it has practical potential in a business context. The app is only the final result of a structured interconnection between the tools provided within the Microsoft galaxy, as will be detailed further on.

---

This thesis starts by introducing the impact of big data and artificial intelligence; in the first chapter the domain of physics will be considered, showing relevant applications of these technologies in different fields. Then, in the following chapter, a focus on the business world will instead be made, showing how these transformations also have a quite measurable impact on different realities and how they have impacted the panorama of companies over the past decade.

The third and fourth chapter will be entirely dedicated to the scientific development of a novel, customized machine learning algorithm for object recognition. The former will present the pre-preprocessing operations needed to prepare the data for analysis, also called feature extraction phase, while the latter will focus on the implementation of the classification, with the definition of a classification template and of different metrics to be compared in order to find the best-performing one.

Finally, the last chapter will be devoted to generalizing the classification model to a real application, dropping the assumptions that characterize the first part of the study. As a result, this time more advanced models will need to be employed; the analysis will gravitate around algorithms employing a single-shot approach and in particular on the YOLO model, a cutting-edge implementation in this respect.

# Chapter 1

# Artificial intelligence in physics

Over the most recent years, a swift and to some extents unexpected progress has been made in the domain of Computer Science and in the technologies behind it. The past decade in particular has seen the steady application of revolutionary ideas like Artificial Intelligence (AI), which is a particular way of programming a computer to model intelligent behavior so that it can learn with minimal or no human intervention, Machine Learning (ML), which refers to the automated detection of meaningful patterns in data, and Big Data, a broad expression to label amounts of data so substantial that can not be processed in traditional ways.

These concepts, once little more than buzzwords to the non-experts, have permeated the daily lives of many, finding fruitful applications across a huge variety of fields. However, this transformation has found an especially fertile ground in Physics, a discipline which has always deeply interlaced the study of data – to be meant in its most "classical" form – with the necessary technological tools to retrieve, process and analyze it.

---

In this Chapter, a focus will be made on the application of these ideas to the physics domain. While not delving into technical details, several examples of such automated algorithms will be presented of such automated algorithms, both to understand their computational power and to appreciate their flexibility in concrete scenarios.

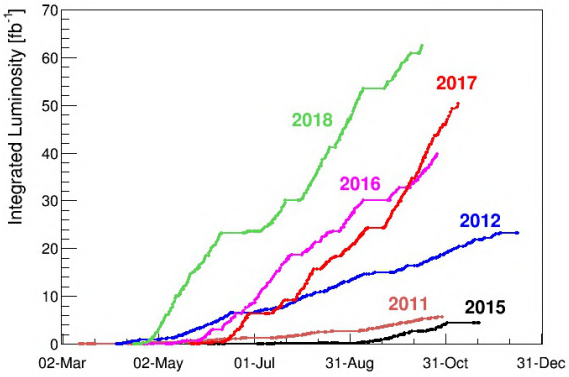
## 1.1 High Energy Physics

As anticipated above, Physics and Computer Science have established a symbiotic relationship from which both have benefited: the former, with powerful and revolutionary models and calculations which have been requiring increasingly powerful computational power, the latter, with the boost to implement the solutions to those very problems into algorithms and to find different, more concrete applications, often pushing them at unseen performances.

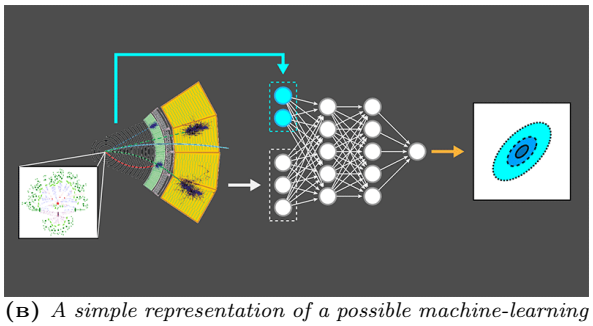
Such a dynamic can be noted across the fields of physics, from cosmology to complex systems, but it happens to be especially relevant for High Energy Physics, devoted to the study of the fundamental building blocks that constitute matter and radiation. Particle physicists, especially in the modern era, have had to face challenges that could naturally be addressed taking advantage of the computer science toolbox: huge amounts of data, critical data-taking conditions, need to extract maximal information are but the most pressing examples [1]. In short, as it was in the past for hardware progress, in a way particle physics has played a catalyst role for advanced software development for physics as a whole.

A possible way to quantify how impacting the increase in data collection has been in recent years in this field is for example by considering the evolution in the amount of collisions detected at the Large Hadron Collider (LHC) at CERN. A 27-km ring in circumference, the Large Hadron

Collider, located in the outskirts of Geneva (Switzerland), is to this day the largest and most powerful accelerator ever built. In it, beams of protons are accelerated until they reach very high energies and then they are made to collide. The byproduct of these collisions is an enormous number of various subatomic particles, which are selected, located and tracked by dedicate hardware for future analysis – each collision is commonly referred to as a *collision event*.




(A) *Trend of the integrated luminosity achieved at the LHC accelerator at CERN over different years of data taking.*



(B) *A simple representation of a possible machine-learning approach for the search of new particles in HEP: from left to right, simulations of particle collisions are used to train a (deep) neural network, allowing for faster measurements of the particle properties – in this specific case for effective field theories (image freely adapted from [2]).*

**Figure 1.1**

The *integrated luminosity* quantifies the number of events observed across a given period of time (the full yearly data taking in this case). This means that the integrated luminosity gives a measure of the total number of collisions that have happened over the integrated time interval, which can naturally be converted into an estimate of the amount of data made available by the accelerator. To have a mean of comparison, 1 $\text{fb}^{-1}$ of integrated luminosity roughly corresponds to $8 \cdot 10^{13}$ – that is, 80 million million – collisions taking place.

As it can be seen from Fig. 1.1a, continuous improvements in the physical design of the LHC have led the collider to increase its integrated luminosity by a factor 10 over seven years and plans are on their way to further grow this value by an additional tenfold. This hardware-wise improvement has clearly had to be accompanied by a corresponding development both from the data management and from the data analysis standpoint. This is why recently an extensive use of artificial intelligence has been made and more and more of such algorithms have been employed to take the most advantage from the increasing amounts of data available, in order to maximize the volume of useful information extracted while keeping the computational burdens under control [3, 4]. This will become clearer in the following section.

### 1.1.1 Event selection using ML

One of the most delicate steps in data collection in the context of accelerators is the *event selection*. Collisions in the LHC generate particles that often decay in complex ways into even more particles, but only a tiny fraction of them are so-called exotic particles, meaning those which may be valuable for possible new physics discoveries. Because extremely large and information-rich data samples are produced constantly (up to about 1 billion particle collisions can take place every second inside the LHC experiments detectors), it is just unfeasible to process, store and analyze all of them; to have a perspective, even 30 million events per second would amount to a total of 2,000 petabytes to store a typical 12-hour run and an average running year would total almost 400 exabytes, an unfathomable amount of data. Luckily, though, it is rather unnecessary to retain all this information anyway.

Historically, trigger systems have always been in place to filter potentially useful data: a trigger is a data-reduction scheme executed in real time – it can be either hardware or software

– that uses simple criteria to rapidly decide which events in a particle detector to keep. On average, about 1 in 100,000 events is kept for future analysis; nonetheless, the amount of data stored at the CERN Data Centers every year is in the order of the petabytes [5]. The use of machine-learning techniques, however, is revolutionizing how humans interpret these data samples, greatly increasing the discovery potential of present and future experiments. Supervised methods can be used to train classifiers that separate signal from background. However, such classifiers cannot be trained on real accelerator data because the true labels for such events are supposed to be unknown. Instead, they are trained with data drawn from Monte Carlo simulations, which generate events and model the interaction of their decay products with the detector [1].
These simulations, repeated in batches of millions, are able to convey a statistically faithful representation of the physics behind a given collision phenomenon, so they can act as an effective proxy for the real events during training. These ones, in turn, fewer in number, will constitute the training set to possibly find interesting physics.

In the particle physics community, there has been a growing effort to use the full high-dimensional feature space to train cutting-edge machine-learning algorithms, such as deep neural networks (DNNs). Deep neural networks are made of several computing units, called *neurons*, distributed into layers in architectures whose complexity can be increased ad libitum. After the mentioned training, these units, not unlike human neurons, are capable of "learning" the assigned problem in its full complexity, therefore ending up performing predictions taking into account a way bigger number of the many descriptive features which characterize a collision event. In particular, DNNs trained at CERN to discriminate between signal and background classes need no manually constructed inputs and yet, according to the most recent analyses, they improve the classification metric by as much as 8% over the best previous approaches [6].

Powerful selection algorithms are also Decision Trees, which employ a tree-like model of decisions to perform classification or regression tasks. These models take a set of input features (represented by the branches) and split input data recursively based on those features, getting to conclusions about the item target value (represented by the leaves); the splits are created recursively and the process is repeated until some stop condition is met. The advantages of this kind of models are their intuitiveness and their ability to learn which selection features are most relevant, which is usually done by assigning a certain cost (computed with specific cost functions) and trying to minimize it. In their boosted version (BDT), many decision trees are combined in a single strong learner. This means that each tree is dependent on prior trees and the algorithm learns by fitting the residual of the trees that preceded it. Thus, boosting in a decision tree ensemble tends to improve accuracy thanks to a supplementary "retroactive" training on previously mislabeled instances. In the LHCb experiment at CERN, a BDT was used which has greatly improved performance while satisfying the stringent robustness requirements of a system that makes irreversible decisions [3].

## 1.2 Medical physics

Medical Physics is the application of physics to healthcare: physics methodology and techniques are employed in a medical context for the prevention, diagnosis and treatment of diseases in patients.

By its own vocation, this branch finds itself at a crossroad of many different fields. Particle physics is found once again, thanks to the development of hardware capable of translating fundamental principles into patient-targeted applications. A few examples are X-ray scans (broadly employed in medical practices like radiography, tomography, fluoroscopy, etc.), magnetic resonance imaging (MRI) and other more sophisticated techniques such as proton emission tomography (PET) or single photon emission computed tomography (SPECT).

Moving on, biophysics, biology and chemistry all provide the main descriptive language

for model building, while the mathematical formalism and theoretical results borrowed from statistical mechanics are often employed. Last but not least, an embedding with engineering and robotics is present as well, especially in the design and production of context-specific tools.

It should hence come as no surprise that such a lively applied area of expertise has opened itself to the revolution embodied by artificial intelligence, machine learning and new technologies in general. As noted in [7], this process, which led experts to coin the expression *e-health*, basically happened on two levels:

- on a physical level, with the development of hardware capable of turning physics principles into patient-oriented applications. A prime example in this sense are *carebots*, miniaturized and increasingly sophisticated robots employed both in the delivery of care and as surgery assistants. Different devices, such as smartwatches or other wearable technology items, have also started to take advantage of their capability of taking and analyzing increasingly bigger amounts of data devoted to monitor the state of health of a person and, in specific instances, even the effectiveness of some treatments [8].

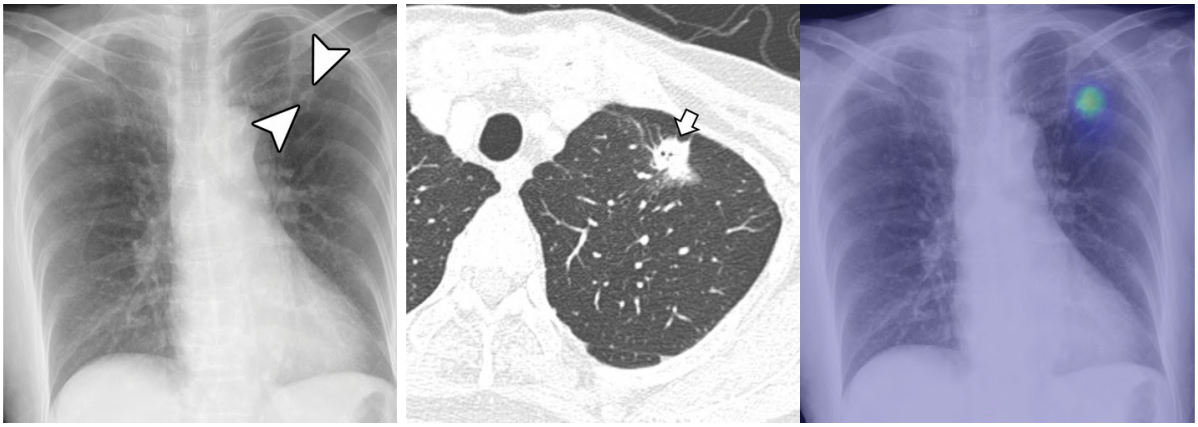
- on a virtual level, there exists an astonishing spectrum of applications whose common denominator is the exploitation of new, powerful and automated algorithms together with the computational power they carry along. Such applications vary from the unsupervised protein–protein interaction algorithms that have sped up decade-lasting research in the field [9, 10] to novel computational methodologies developed to identify DNA variants in order to predict and deal with hereditary diseases.
  Virtual medical applications of AI, though, do not pertain just to high-level research scenarios. An increasing number of patient records in general medicine, for example, has been digitized and powerful automated algorithms have been employed to find meaningful patterns across millions of data points and categories. Furthermore, trained ML models have been increasingly aiding doctors in the diagnosis process, often with tangible results that often surpassed human performances [7, 11, 12].

While it will not be a focus point in this thesis, it is indeed worth pointing out some of the ethical tolls connected to the phenomena described in this section, especially considering their field of application: the exponential spread in the use of data to monitor patients [13], the capabilities of more and more everyday-devices to store and process such information and the delicate matter connected to the sharing of such data with the manufacturers [14], as well as the very fact that devices and chips now have the capability to be fully integrated in the human body itself, thanks to the impressive recent progress in miniaturization [15, 16]. These factors have all necessarily implied deep and complex ethical considerations, which are far from being univocally answered.

### 1.2.1 Image classification in medical physics

Now a focus is made on the applications of image analysis performed in medical physics and medicine in general, as yet another declination of the application of computational discoveries to a physics-related subject in mostly concrete cases.

In [17], Nam and others describe the insightful application of Deep Learning-based Automatic Detection (DLAD) to the context of chest radiology and the detection of malignant pulmonary nodules. In this study, $43,262$ radiographs were considered, $9,225$ of which were in fact displaying pathological conditions from the patients. A semi-supervised approach was chosen, meaning that a portion of the nodules displayed in the radiographs were manually labeled to train the algorithm. The dataset was then fed into a 25-layer neural architecture which had both a classification and a localization task to perform. The outputs are probabilities that there would be a nodule in each sample.



(A) (B) (C)

**Figure 1.2:** *Representation of a chest radiograph sampled by DLAD. In (a), a 1.9-cm faint nodule is pointed by the arrows, which nodule was detected by only 11/18 observers; in (b), a CT examination confirmed its malignant nature; in (c) the individuation performed by DLAD, which correctly underlines the presence of a possibly pathological feature, making it more blatant to human observers (ending up in 5 more radiologists being able to detect the nodule).*

The algorithm showed high specificity, being able to detect 100% of high conspicuity nodules; also, a sharp decrease in the rate of false positives was observed, while the sensitivity was preserved. DLAD ended up outperforming 16 out of 18 professionals (selected for the experiment so as to have an interval of experience in the field ranging from 1 to 26 years); this remained the case even when traditional computer-aided diagnosis techniques were used as support.

What is more important, the radiographs processed with DLAD were much easier to interpret by the experts. This can be directly appreciated by considering Fig. 1.2, where a faintly-visible nodule in a radiograph (a) – confirmed to be lung adenocarcinoma by a contrast-enhanced CT examination (b) – was correctly pinpointed by the automated algorithm, resulting in 5 more radiologists being able to identify the feature and 8 seeing their confidence increase. The biggest limitation for the algorithm was the one connected to human perception, as it was necessary to provide the algorithm with pre-labeled samples for its training stage.

Another application close to chest X-rays was explored in [18], particularly in connection with COVID-19 patients. This study has shown that AI- and radiologist-assessed disease severity scores on the chest x-rays (CRXs) analyzed comparable predictors of adverse outcomes in patients with COVID-19. It was also evidenced that the initial x-ray severity assessed by a deep learning AI system may have prognostic value in COVID-19 patients, again with a performance comparable to a radiologist-assessed score.

The comparable performance of the AI system with respect to a radiologist-assessed score in predicting adverse outcomes could represent a game-changer for resource-constrained settings as the COVID-19 pandemic keeps spreading, especially for those countries with a shortage of radiological expertise. The possibility of having the lung disease severity rapidly assessed by an AI system, together with patients' clinical data, may help medical teams identify patients at a higher risk of an adverse outcome straight at the first aid presentation and thus allocate the limited resources more efficiently. The main limitation of this study is the retrospective design, which can lead to observer bias.

This is far from being the only impacting application of machine learning in the fight against the COVID-19 pandemic; a particularly significant example can be found in the AI-SCoRE (acronym of Artificial Intelligence - Sars Covid Risk Evaluation) project. This is an autonomous learning platform capable of calculating for each individual - based on a series of clinical and diagnostic indicators - the probability of developing the most severe forms of

Covid-19, thus allowing targeted and timely health interventions and reducing the impact on the health system [19].

The aim of the project is twofold: on the one hand to recognize in the general population the people at greatest risk of developing severe forms of COVID-19, if infected with the virus, that should be protected the most; on the other hand, to recognize among the patients who show the first symptoms of COVID-19 those who will have the worst prognosis. The project is thought to start from this second objective, with an AI algorithm that will integrate diagnostic images, clinical and laboratory parameters, inflammatory status, and genetic profile of the patient and the virus.

Conceived by researchers at the San Raffaele University who have provided data from 2000 patients for training, this endeavour has led to a partnership with two world giants of information technology such as Microsoft and NVIDIA, capable of providing the infrastructures needed to develop the computational side. In particular, Porini, center of excellence and international partner of Microsoft on Cloud Azure platforms, conveys the technological platform based on the latest Microsoft innovations for data analysis and artificial intelligence [20]. This will allow the collection, processing, management and use of heterogeneous data, coming from multiple sources, in total respect for privacy, to provide medical and research personnel with timely and detailed information to support the decision-making phase and the processes necessary to respond to the various phases of the emergency.

This project is the first of many virtuous examples of the results that can be achieved with a collaboration bridging the research world with the business one.
In some respects, this kind of approach also represents an initial step toward the model of personalized and precision medicine which will reasonably characterize the way humans will interact with the health system, thanks to the amount of data at disposal and the computational tools to analyze it.

## 1.3 Physics of Complex Systems

The expression *complex system* commonly describes a system made of many components, which may or may not interact between each other. Such systems may differ substantially (recurrent examples are the Earth's climate, communication infrastructures, social networks, neuronal activity), but their common feature is that they are intrinsically difficult to model. In short, their behaviors cannot be easily inferred from their properties and the complex dependencies that arise make it impossible to consider them just as a union of their individual parts; so much, in fact, that it is often the case that they give birth to otherwise unobserved phenomena, which are not apparent from its components in isolation – this peculiarity is referred to as *emergence*.

The physics of complex systems is devoted to the study of these ensembles. Because most of them are made up of entities in the order of the billions or more, it is computationally unfeasible to study such a system and its evolution from the perspective of each individual; this required the development of different approaches and mathematical techniques. Unsurprisingly, a broad variety of concepts was borrowed from traditional fields of physics like thermodynamics, where a similar comprehensive description is required (an ideal gas, for example, is hardly approachable if considered as the sum of its particles, while it shows interesting properties when treated as a whole).

The advent of AI and ML-based algorithms, while not entirely solving the computational burden that these systems embody, definitely allowed for new perspectives to be explored. Machines have time and again proven to be able to detect new patterns, to perform more accurate predictions taking into account a much higher number of descriptive features and also to be able to efficiently deal with sets of data orders of magnitude more populated. Not unlike the theoretical change in perspective required to consider complex systems as "separate entities" instead of a mere combination of their parts, so are automated algorithms now allowing

researchers to tackle long-standing problems with new rules defined by these computational innovations.

For example, several studies have addressed the challenges connected to condensed-matter physics [21], which is the study of the collective behaviour of infinitely complex assemblies of particles: sophisticated machine learning approaches are capable of describing collective interactions between many atoms or bonds combining accuracy and efficiency ([22]), other architectures, such as fully connected and convolutional neural networks, can identify phases and phase transitions in a variety of condensed-matter hamiltonians (in [23], for instance, the renowned Ising model is considered); finally, several researches have also dealt with quantum matter, studying phases of matter whose properties are intrinsically quantum-mechanical (strongly-correlated systems, superconductors, ...) [24].

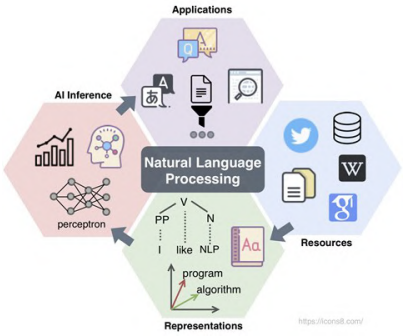
Moving away from purely physical research cases, a lot of interest has been drawn to the complex task of weather forecasting, turned into a data-intensive prediction to be performed mostly employing deep learning ([25, 26]) but also simpler regression models ([27]); neural networks have proven to be valuable allies also in studies of social media, either to model the behaviors of the people interacting in it [28], or with more immediate practical use such as spam or fake news detection [29]; last but not least, several AI implementations can be found in the recent projects for the development of smart cities, urban areas where traditional networks and services are made more efficient with the use of digital and telecommunication technologies – mostly thanks to the huge amount of data being collected and to the algorithms capable of properly processing it – for the benefit of its inhabitants and businesses [30, 31, 32].

### 1.3.1 Natural Language Processing (NLP)

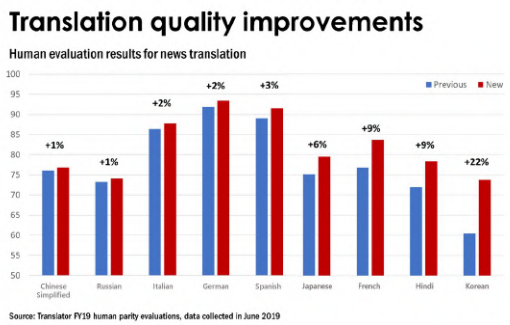
A particular insightful application, which allows to mention yet another sort of automated algorithm, is the one connected to the sphere of Natural Language Processing (NLP). This expression refers to a field of AI that gives the machines the ability to read, understand and derive meaning from human languages. The ultimate goal is to obtain a software which is capable of actually understanding language-based interactions – a difficult task in and of itself, which is further complicated by its impalpable definition –, from which a multitude of auxiliary applications stem (information extraction, automatic document organization, speech recognition and many others).

The dominant challenge in NLP is that human language, while having some background rules (mostly connected to grammar, morphology and semantics), is a flexible entity, filled with nuances, unspoken long-term references and hidden connections. It is a prime example of a source of *unstructured data*, data which lacks a pre-organization in a defined manner and it is hence messy and hard to manipulate. Historically, theoretical and computational approaches to language modeling were based on the attempt to interpret a text or a speech based on its keywords. This is known as *mechanical approach* and it is exemplified for instance in [33], where physical concepts like entropy and long-range correlations are employed to find meaningful patterns in two English works of literature. In most recent years, this perspective was replaced by the *cognitive approach*, which is based on actually understanding the meaning behind the words; the training happens through large corpora of real-world examples, often extracted from the Internet (Wikipedia pages, tweets or posts from social media and so forth), so that it is even possible to detect figures of speech like irony, or even perform sentiment analysis.

This is the founding mechanism of Recurrent Neural Networks (RNNs), which exploit a set of time-delayed feedback connections to store past information, necessary to fully interpret the complexity of written texts. Their input is no longer a static vector, but rather a time series of ordered vectors [36]. A further sophistication can be found in Long-Short Term Memory (LSTM) networks, which can learn how much information should be retained in temporary memory and how forward it should be propagated [37, 38]. Automated learning procedures




(A) *A schematic summarizing the stages revolving around Natural Language Processing, from the retrieval of data from a multitude of possible sources to the formalization of ML algorithms and, finally, the creation of a broad range of day-to-day applications.*




(B) *Comparison showing the improvements registered in news translation from English to various languages thanks to the employment of ML-based architectures. The evaluation scores are obtained on a standard translation task; further discussions about experimental methodology can be found in [34] (plot freely adapted from [35]).*

**Figure 1.3**

make use of statistical inference algorithms as well, in order to produce models that are robust to unfamiliar inputs, such as words or expressions that have not been seen before and to erroneous inputs (e.g.: misspelled words or inappropriate vocabulary).

An illustrative example of the leap undertaken by NLP thanks to artificial intelligence can be found in online translators, which, in the past decade, have switched to this approach instead of having their software translate portions of text by using a fixed set of rules. In particular, Fig. 1.3b shows the improvements in several languages achieved by neural machine-translation models developed by Microsoft, as evaluated by a panel of human translators on a standard translation text. It should be noted that, aside from French, the highest upgrades were seen in exotic languages such as Japanese (+6%), Hindi (+9%) and especially Korean (+22%). Unsurprisingly, these are all languages with such a high degree of complexity that conventional, head-on approaches are destined to fail and only the computational capabilities and the analysis power of new algorithms made it possible to obtain human-like interpretation performances [34].

As far as speech recognition is concerned, Amazon's Alexa, Apple's Siri, Google Home and Microsoft Cortana are the most immediate examples of intelligent voice-driven interfaces that use NLP to respond to vocal prompts and perform an incredible variety of tasks at the user's request. Finally, also most of the applications already cited have some sort of connection to NLP (fake news detection for example, but also the analysis of patient records mentioned in medical physics).

# Chapter 2

# Artificial intelligence in the business world

TOGETHER with the advent of AI and machine-learning techniques, the big data revolution has hardly been confined to the world of scientific research. Especially in the past 10 years, in conjunction with a widespread interest across several fields of studies, these concepts started to permeate into the real world, at first involving companies either related to software development or connected to very narrow, case-specific applications; then, one advancement after the other, these novelties broadened their range of employment, hence reaching a much wider community.

---

In this chapter, AI applications in the industrial reality will be considered. In particular, an overview of the fourth industrial revolution will be presented; its implications will be studied especially for the so-called small and medium enterprises (SMEs); then, the phenomenon of big data as business model will be examined and this will give the opportunity to consider a specific example, that of Porini Srl, where this work sees its first light.

## 2.1 The fourth industrial revolution

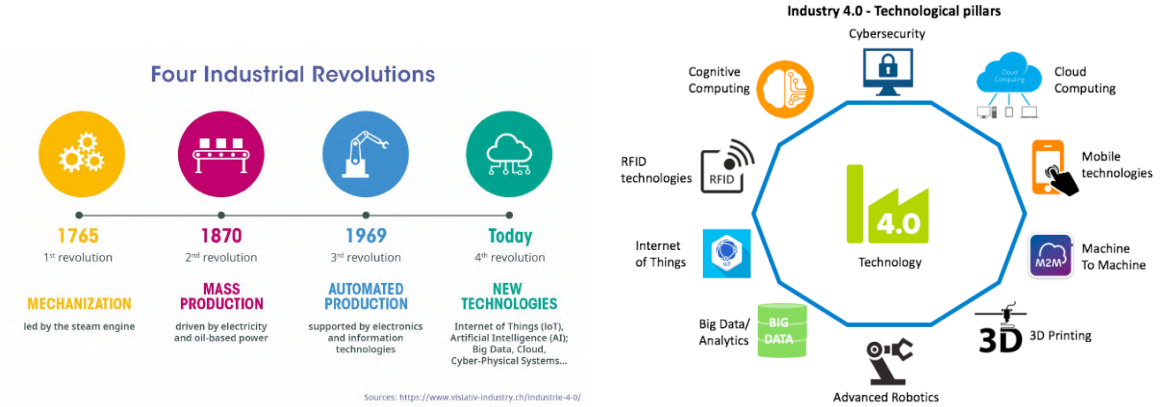
In the context of research, big data and machine automated algorithms to process it are often employed with a *curiosity-driven* approach: this means that a practical, result-oriented mindset can be at times overlooked in favor of the exploration of different computations techniques, ways to optimize them or even simple attempts at cross-fields applications. Clearly, this does not imply a complete lack of practical and often fruitful applications, many examples of which have been explored in the previous sections, but it underlines how many pragmatic aspects may be neglected for research purposes.

The academic workflow is hardly replicable when dealing with businesses, private-sector industries or similar entities; in all these instances, an *application-driven* approach is instead dominant. The employment of these new technologies, not unlike any other industrial strategy, needs to be connected to tangible, preferably short-termed benefits, both in terms of economic revenues and in the development of products or services which will end up providing that specific business with an edge over its competition.

A significant difference lies also in the characteristic timescales of these two worlds. Universities and research entities do indeed have to account for a budget and deciding to pursue the promises of big data analytics certainly represents a sort of gamble (for example, in the decision to open a new department and invest in specific faculty training). However, they can do so with the capability of bearing costs that will pay dividends in the long run. This just can not hold in

the business world, where dramatic changes happen at all company scales in way shorter time windows, especially in nowadays markets, which are more dynamic and unpredictable.

These caveats should not lead anybody to think that the big data revolution has not had a profound impact on the private business sector as well. In fact, the opposite is true: the industrial world has undergone tremendous changes in the past decade, mainly connected to what is being called the *fourth industrial revolution* (Fig. 2.1a). This expression describes the advent of cyber-physical systems involving entirely new ways in which technology becomes embedded within societies and even our human bodies [39, 40]. The concrete realizations of this concept are those companies where data are being produced and used in a virtuous and vibrant circle of collection, storage and insightful analysis. In this scenario, new technologies and interconnected devices perform the first two tasks, while ML-based software and AI are dedicated to the last one. Needless to say, the new automated algorithms, together with the stunning improvements in artificial intelligence, have boosted this process, creating new opportunities as well as new challenges in the variegate business sector landscape [41].




**(A)** *A stage-by-stage representation of the four industrial revolutions, with the core concepts that acted as catalyst for each.*

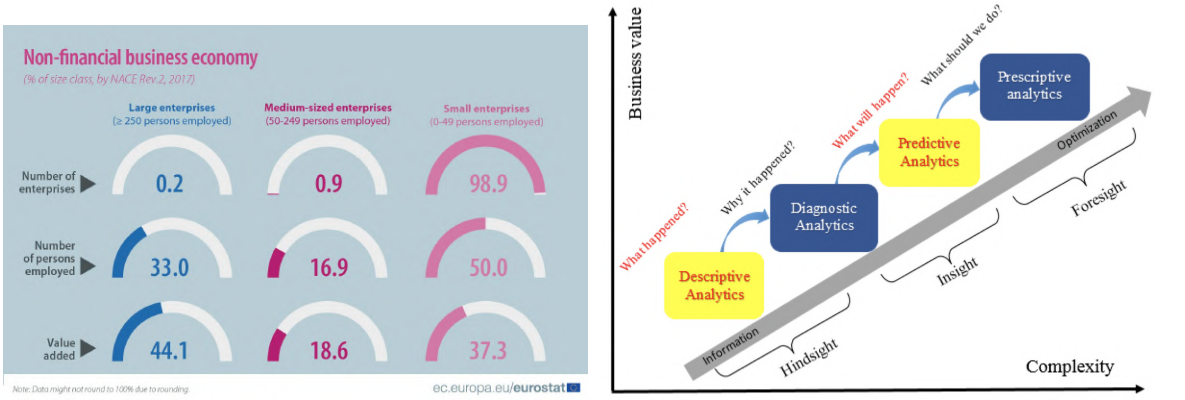
**(B)** *The main technological ideas gravitating around the fourth industrial revolution. As it can be seen, big data, artificial intelligence and IoT are just a piece of a very articulated puzzle (image freely adapted from [42]).*

**Figure 2.1**

This industrial revolution has once again profoundly shaped the markets and their players, mainly affecting their practices and the tools at their disposal [43, 44]. Still, the effects have also become cultural, in that todays industry world is a reflection of the everyday reality, with its interconnected devices which generate continuous flows of data that are stored and analyzed by intelligent algorithms for whatever purpose. As seen with any other industrial revolution, sudden changes inevitably entail instability and this is even more true nowadays; digitalization, AI, big data and others often prove to be invaluable keys to open thriving possibilities to businesses. Today, this is the power of seeking information in data.

## 2.2 AI for Small and Medium Enterprises

The expression "small and medium enterprises" (SMEs) – also referred to as small and medium businesses (SMBs) – is an umbrella term which is commonly employed to group businesses with less than 250 employees and a turnover not exceeding €50 million. It is estimated that as much as 99% businesses in EU fall under this definition, employing approximately 66% of the total workforce; instead, in emerging economies, SMEs represent 60% of the total employment force, contributing as much as 40% to the GDP of these countries [45] (see Fig. 2.2a for a visual comparison). This broad categorization makes SMEs the most populated kind of business: it should come as no surprise that they are chosen as a representative case study to analyze the




**(A)** *Business subdivisions according to their size in terms of number of enterprises, people employed and contribution in economic value in the European Union.*

**(B)** *The four types of data analytics employed in the industrial sector, scaled according to their value and degree of complexity; the question each tries to address is also highlighted (image freely adapted from [45]).*

**Figure 2.2**

use of AI in industrial contexts, both in this thesis and in literature.

In 21st-century enterprises, there exists continuous streams of data, coming from an ever increasing multiplicity of sources. SMEs may get datasets from their email, business records, company website logs and activities (either internal or from their customers), online reviews, social media marketing, just to cite a few examples [46].

Moreover, over the past few years, the so-called revolution of the Internet of Things (IoT) took place. The term refers to the extension of computing and network capabilities to devices and sensors that are not considered as computers, but that become capable of making machine-to-machine interactions with minimal or no human input. At the industrial level, this means that there exists the possibility to collect and store any kind of data, developing production monitoring systems which make it possible to quantify virtually all the aspects of the processes happening within a business. IoT devices retrieve data and create an interconnected network, but it should be clear that they often lack the capabilities of performing analyses themselves, especially those tailored to the specific needs of an activity [47, 48].

In this framework, businesses could encounter huge value in data and their use for decision-making. A brief review of the four main concrete applications of big data analytics is displayed in Fig. 2.2b, where they are ordered in terms of their business value and their complexity. We now itemize some of the areas in the industrial sector where the biggest advantages from big data applications can be seen:

- *production optimization*: the availability of unthinkable sets of data, which can also be put in connection one with the other, is a great resource to closely monitor processes and to hence be able to elaborate well-informed strategies and reduce long-term operational costs;

- *forecasting*: big data combined with AI and automated algorithms have powerful prediction powers, which are way more advanced than common analytics techniques. This means businesses could end up having the capabilities to strengthen their prediction thanks to both an increased statistical robustness (more data available) and an incredibly higher analysis power (more powerful algorithms), greatly mitigating risks;

- *marketing optimization*: the aforementioned strategies can be turned to benefit the marketing compartment as well. Just like in production, commercial strategies can highly benefit from big data about customer profiles and habits in order to improve products or services perception on their share of the market;

- *customer acquisition/retention*: data-driven commercial strategies can also serve the specific purpose of exploring the customers needs and requirements, both to consolidate the business pre-existing share and to identify new potential markets. Big data allows narrower segmentation of customers and therefore much more precisely tailored products and services;

- *productivity increase*: companies that decide to switch to the use of these new technologies see their overall productivity soar; again in [45] it is noted how data-using firms in the top quartile are 13% more productive than those in the bottom quartile.

Such extensive and rooted advantages are nonetheless somewhat the end step of a process which may turn to be long-lasting and tortuous, especially for SMEs. Digital transformation is requiring companies to rethink and innovate their business models, which can easily turn out to be a disruptive process especially for small and medium businesses. Even in the multifaceted context of SMEs, it is possible to outline what the most common and pressing challenges are in undertaking the transformation towards becoming a fourth-generation industry [49].

- Costs are generally the biggest challenges for SMEs, which may find themselves in the situation of not being able to afford the inevitable expenses needed to revolutionize their whole productive compartments. This is especially true because, while offering great strategies for future economic improvements, the digitalization process, not unlike any other investment, indeed requires the availability of immediate liquid assets to afford technology and set of knowledge, both expensive addictions.

- Time is another great limiting factor, also because in the business world it is often tightly connected to costs. Smaller entities have difficulties in giving up short-sighted profits even in exchange for long-term benefits, especially taking into account that this choice often requires the disruption of ingrained practices in ever-competing markets. This is another clear example of the strong caesura with the academic world, where neither of the two mentioned factors represent a defining issue.

- There exists a widespread, cultural brake that leads especially smaller enterprises to blindly rely on conventional business methods. The intuition-based approach, based on the ability of a single individual (often the company's owner) or a restricted group of people (its managers), is long-standing and hence perceived to be safer even from an historical point of view. However, the final result is the missed opportunity to implement a data-driven business, which is bound to be more effective, accurate and statistically robust than human-based decision-making.

- Connected and in fact complementary to the above, there is a generalized low awareness of the data value and potential. The mantra *data is the new oil* seems to have caught the attention only of a restricted group of enterprises, usually the bigger and more tech-oriented ones. Data is information and as such it represents an asset in and of itself, regardless the even more profitable internal applications it could be also employed for. However, most SMEs still appear not to be tempted to make use of business data other than for record-keeping.

- The analysis of business data often requires a set of knowledge which can not be limited to the sphere of big data analytics or the simple implementation of machine learning algorithms. The ideal professional needs to combine these technical skills with domain-specific knowledge, granting deeper insights on the data that is being studied and the most cost-effective ways to use it for the enterprise future growth. This kind of hybrid profile is still somewhat rare and in high demand, making it a further challenge for smaller or less advanced businesses, especially those in emerging countries.

- A poor knowledge of the available public funds represents a further obstacle: while most states, as well as the European Union, are providing incentives to push SMEs toward the path of digitalization and to realize important innovative digital and data-driven ideas, only a small fraction of SMEs seem to be aware of these opportunities and actively use them to facilitate their own transition process.

## 2.3 AI as a business model

Like any other industrial revolution in history, the digitalization process has proven to be a disruptive one and it has been stressed how this disruption is more impacting especially on small and medium businesses. However, as it has been pointed out, the advantages of a data-driven, AI-based industrial approach are multiple and tend to respond to a cost optimization strategy in the long haul.

While it is reasonably incorrect to assume that all SMEs cannot afford sophisticated computer systems, initial costs both in terms of infrastructures and dedicated expertise indeed play a critical role when a company is considering to undertake a digital transition. SMEs often do not have in-house capabilities for the selection, installation, configuration and maintenance of complex IT systems, which constitutes a potential barrier to big data analytics. According to [50], an online survey conducted between 15 Welsh manufacturing companies showed that while as much as 80% maintain data about their customers and only 7% feel they have difficulties in data storage, a staggering 53.3% claims to be unaware of big data and, among the others, only three quarters had a clear vision of what they would hope to achieve and how. Although this is a limited sample, a similar situation is also portrayed in the already cited [45], where 53 SMEs from UK are instead taken into account. These pieces of data show that the awareness of all that could be achieved through the application of big data analytics and ML algorithms is perhaps not as robust as it could be expected.

In general, most businesses collect and store data but require additional skills to conduct advanced data analyses and produce useful statistics and correlations that can support them in making strategic decisions. All aspects of this process, from data retrieval and processing to storage and use of third-party cloud applications, have been in very high demand, but small businesses still tend to lag behind their larger competitors, especially when it comes to deeper statistical analysis and predictive modeling. It should be unsurprising, in this market context, that these unaddressed demands led to the creation of a new kind of consulting sector, born to provide businesses with these sorts of technological services and products. In practice, new companies have flourished that have made big data, AI, IoTs and similar concepts their core business to be exported according to the needs of their customers.

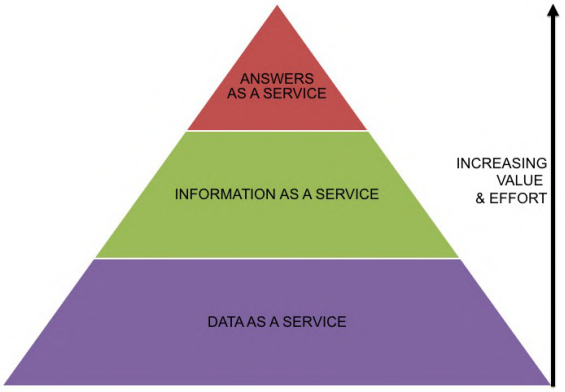



**Figure 2.3:** *A representation of the three consulting business models related to (big) data analytics in the fourth industrial revolution era, arranged according to the effort they require and the resulting value on the market (picture freely taken from [51]).*

Ultimately, the practical outcomes granted by this kind of consulting have several common points with the beneficial results deriving by the digital conversion listed in the previous section; the difference in this case is that a customer-provider scheme is in place. A certain business may be willing to undergo changes to develop some sort of state-of-the-art technological infrastruc-

ture, whatever these changes may be, but it could encounter implementation difficulties related to lack of knowledge, qualified personnel or even dedicated time. The provider represents the entity to which it is possible to outsource these challenges so that they can be brought to completion by dedicated experts in the most convenient and suitable way. As noted previously, the technological assets are but a part of the whole mosaic, which also requires companies who offer this kind of consulting to have a deep understanding of the businesses they cooperate with and the different ways they fit in the market.

This digital consulting can be theoretically articulated in three distinct business models, according to the needs of the customers and the solutions they require from their providers [51]. The first one (Fig. 2.3), with the lowest degree of complexity, is known as *Data as a Service* (DaaS): its main scope is to supply large amounts of processed data, also in raw forms, with the idea that the customer's job will be to use it to boost performances or to implement the best strategies for their own buyers, according to their needs. In the second kind, referred to as *Information as a Service* (IaaS), there is a focus on providing insights based on the analysis of already-processed data. In this case, there is an additional layer of complexity, given that the customer now expects to have conclusions of what needs to become a precise strategy within a well-studied market case. IaaS customers often lack the ability or resources to process and analyze data on their own; rather, they are willing to exchange value for analysis from trusted parties, as the IaaS business model is centered on turning data into information for customers who need something more tailored (clearly at an extra cost). Finally, the third, most complex customer-provide relation can be identified as *Answers as a Service* (AaaS), which is focused on providing higher-level answers to specific questions rather than simply the information that can be used to come up with an answer. AaaS customers often need specific directions in order to make decisions.

As it must have become clear from this broad overview, businesses who find themselves in this slice of consulting are generally selling to other companies services instead of products. These are the companies that have the disposable income and the planning skills to turn digital but may lack the capabilities or are willing to outsource these domain-specific challenges. The result has been an acceleration of the big data economy, which has seen an increase in the number of players and consequently in the applications in which it has been able to find fertile ground [52, 53].

In this panorama, the advent of cloud-based solutions is yet another form of collaboration between different businesses in the digitalization process. A cloud allows to take advantage of solutions that are often developed by the so-called tech-giants, which have enormously higher spending capabilities and knowledge about the infrastructure to be developed. The business, in turn, benefits of unprecedented computational power and storage capabilities, which could hardly be developed individually. A cloud-based system also offers other crucial advantages, granting much lower costs compared to in-site solutions (which would need to be planned and realized specifically for that business and also require expensive upfront hardware-related costs) and avoiding the need to specifically train personnel to develop these solutions (which, as mentioned, requires ad hoc professionals whose costs are often unbearable for small or medium companies). Large scale solutions, moreover, also grant higher standards in terms of security protocols, a particularly sensitive matter in the industrial world.
On the other side, however, the transfer of data to external servers can of course pose issues to certain kinds of businesses, especially pertaining to the storage of sensitive information. As with other instances, outsourcing infrastructure also implies the necessity to trust third-party applications and to partially lose control over the generated data.

This sort of business relationship indeed serves the purpose of showing how in the digitalized era several examples exist that prove the possibility of a symbiotic collaboration between the so-called tech giants and smaller companies. The former have unparalleled resources and technological infrastructures whose specifics can hardly be matched, but the latter tend to have

a greater knowledge of their local realities or in some specific fields of application among the many that see in digitalization a key for their future evolution. The net result is a flexible industrial solution tailored to the customer's needs, which however also relies on safe and world-class architectures that only few tech top players can actually grant.

### 2.3.1 A concrete example: Porini Srl

Porini is a competence centre and a Microsoft Gold Partner with the mission to support the management of medium and large enterprises in Italy and worldwide, in the design and implementation of excellence solutions that can accompany all business functions towards the digital transformation in progress.


**Figure 2.4:** *The Porini Srl logo.*

Porini's suite of solutions based on the Microsoft platform and the Azure technology is available on premise and in cloud and it is offered worldwide both directly and through a network of qualified partners.

With a team of about 160 professionals located in 4 countries (Italy, Portugal, USA and India) and a turnover of about €13 million, Porini supports its customers in the adoption and development of solutions to improve the decision-making systems and governance of the company, using technology and innovation to equip themselves with the appropriate tools for the achievement of strategic objectives.

Established in Como in 1968 as a company specialized in solutions and consulting services specific to fashion, clothing, textiles and retail companies, over the years Porini expanded its expertise becoming a Microsoft Global ISV (independent software vendor), expanding its portfolio solutions with Social CRM, Business Analytics, Artificial Intelligence, Machine Learning, IoT, Performance Management, Collaboration and Knowledge Management targeted to medium to large companies in many other industry sectors like Manufacturing, Financial Services, Travel & Transportation and Health. Since 2018 Porini has become part of the DGS group: thanks to this agreement, Porini is even more a centre of competence on Microsoft technology platforms, both nationally and internationally, focusing on the manufacturing, retail, fashion and textile sectors.

This thesis takes its first steps in this prolific business context. A case study will be presented and thoroughly discussed in the following chapters; this problem reflects a real-life scenario like the many tackled daily at Porini and it will show how big data and the implementation of advanced automated algorithms are but a component in the study process. They need to be accompanied by planned analysis, a flexible use of methods from different fields (several instances from physics have been previously explored) and a knowledgeable application of specific techniques.

# Chapter 3

# Pre-processing and color analysis

Artificial intelligence algorithms have deeply impacted both the research and the industrial world, bringing along a swift and marked revolution. They significantly outperform previous standard techniques and their flexibility makes them valuable for virtually any sort of application.
In the industrial world, in particular, they have contributed to a deep transformation of the business models, especially in SMEs. The new data-taking capabilities, integrated with these powerful tools for analysis, have allowed a digital transformation that has been changing most businesses, both in their internal structure and in the tools at their disposal to perform their activities.

Among the extremely diverse practical use cases, ML algorithms can be employed in the large-scale retail trade for product recognition or goods control. This scenario has several useful applications, such as systematizing the management of large stores, performing commercial analysis in an automated fashion, but also providing customers with product information and aiding them during shopping [54, 55]. Object detection and recognition in grocery scenarios is very complex because object appearance is highly variable due to relevant changes in scale, pose, viewpoint, lighting conditions and occlusion [55].
The ultimate goal of the analysis that is going to be presented is the automatic recognition of grocery products, starting from pictures of them taken in standard conditions in supermarket-related contexts. This result will be the outcome of several steps of data preparation, as well as studies and detailed planning of the algorithms to be employed and their underlying models.

To achieve the target of object recognition, even regardless the supermarket context, the immediate idea is understandably to employ the power of deep neural networks (DNNs). However, as powerful and efficient as they might be, such algorithms offer little insight on the data itself, which becomes little more than a collection of labeled images fed into a black box. In order to have a deep understanding of the data under analysis, it was decided to first perform a customized analysis, focusing on the images, their structure and their numerical description. This kind of physics-oriented approach will allow to closely follow the study process, controlling and thoroughly quantifying each stage; this will lead to a broader understanding of the matter, hopefully providing hints to optimize the following classification stage.

To approach this preparatory study, this first part of the analysis will be conducted by restricting the studies to a *toy model* limited to 5 grocery product classes: pasta Barilla, panettone Bauli, Kinder Bueno, Ferrero Rocher and Tic Tac. A sample image for each class is shown in Fig. 3.1.
It should be noted that the selected products show a net variation between their typical color palettes, which in the sample preparation phase was considered to be a key variable to evaluate the quality of a color-based classification against different scenarios. The products all share somewhat of a parallelepipedal shape; an investigation on this aspect will therefore not be part of the present analysis.


(A) *Pasta Barilla* (B) *Panettone Bauli* (C) *Kinder Bueno* (D) *Ferrero Rocher* (E) *Tic Tac*

**Figure 3.1:** *Sample images for each of the considered classes in the color analysis toy model.*

Furthermore, the only images to be considered will be the ones taken in a studio, with a single product on display on a white background. The idea is that in this kind of images the volatility is expected to be under control, at least compared to the notoriously noisy and high-randomness pictures representing supermarket shelves. This will make it easier to grasp what the defining features of each product are and, possibly, to extract and employ them for classification
Having only one product will also allow to focus on the techniques specific of the domain of color analysis; it is also true, on the other hand, that this represents a considerable simplification compared to a real scenario, which may make this part of the procedure harder to generalize.

---

In this chapter, elements of the theory of color will be presented, giving a rigorous yet concise perspective; this will serve as introduction to the pre-processing part of this study, which will be the object of a comprehensive presentation as it represents a founding operation to the success of finer analysis implementations. In particular, the equalization process will be detailed, needed to calibrate the white color in the images, and the cropping routine fundamental to isolate the target product from its background.

## 3.1 Elements of theory of colors

### 3.1.1 Colors and color mixing

In physics, a color is defined as a combination of electromagnetic radiations with a certain intensity and wavelengths usually assumed to be within the visible spectrum of light (for humans, approximately from 390 nm to 700 nm), so that it can actually be perceived as such. Physically, objects can be said to have the color of the light leaving their surfaces, which depends on the spectrum of the incident illumination and the reflectance properties of the surface, as well as potentially on the angles of illumination and viewing and, in some instances, to the light itself that certain objects are capable of emitting. A useful concept in understanding the perceived color of a non-monochromatic light source is its dominant wavelength, which identifies the single wavelength of light that produces a sensation most similar to the perceived color of the light source itself.

While this thesis will disregard physiological and biological aspects of vision and color perception, it should be stressed right away that a color is rather a function of the human visual system, not an intrinsic property of matter. One may formally define a color as a class of spectra that give rise to that specific color sensation, but such classes would still vary widely among different species, and to a lesser extent among individuals within the same species. Color is in fact determined first by frequency and then by how those frequencies are combined

or mixed when they reach the eye. Light falls on specialized receptor cells (called cones) at the back of the eye (the retina) and a signal is sent to the brain along a neural pathway (the optic nerve); this signal is processed by the occipital lobe, found at the rearmost portion of the skull, where the visual cortex and visual processing center reside.


(A) *Subtractive color mixing.* (B) *Additive color mixing.*

**Figure 3.2**

Colors can be formally described according to mathematical models that dictate their representation with a precise account of how the components are to be interpreted. Such representations can in theory have an arbitrary degree of complexity, but they are usually based on the definition of a limited number of *primary colors*, usually a triplet, that can be combined in different proportions to obtain all the other colors and shades. There are two ways to define how two or more colors can be merged: subtractive color mixing and additive color mixing, represented in Fig. 3.2 with the 3 primary colors for each. The former (made by yellow, cyan and magenta) consists of creating a new color by the removal of wavelengths from a light with a broad spectrum; for example, this is what happens when mixing paints or inks in a printer: wavelengths are absorbed (therefore subtracted) from white light and the color that a surface displays comes from the parts of the visible spectrum that remain visible. In the additive case, instead, the primary colors are red, green and blue; a new color is created by joining two sets of wavelengths. Clearly, this is what happens with light: all of the different wavelengths of sunlight create white light rather than many individual colors, meaning that all of the the wavelengths still reach the human eye (they are combined together). Projectors and computer terminals are examples of additive color systems and hence this will be the intuitive frame for this study.

### 3.1.2 Color models

Having defined the three primary colors, there are a number of methods for specifying any given color in terms of these three. By definition, a *color model* is an abstract mathematical model describing the way colors can be represented as tuples of numbers. In essence, a color model is a system that uses three primary colors to create a larger range of colors. There are different kinds of color models used for different purposes, and each has a slightly different range of colors they can produce. When this model is associated with a precise description of how the components are to be interpreted (meaning a specific, measurable, and fixed range of possible colors values), the resulting set of colors is called a *color space*[1]. Its most basic practical function is to describe the capabilities of a capture or display device to reproduce

[1] A color space identifies a particular combination of a color model and a mapping function. In general the two expressions are used interchangeably, but in a strict sense it is incorrect to do so (a single color model can be associated to multiple different color spaces). This ambiguity will occur also in this thesis, but only in cases where no room for misinterpretation is left.

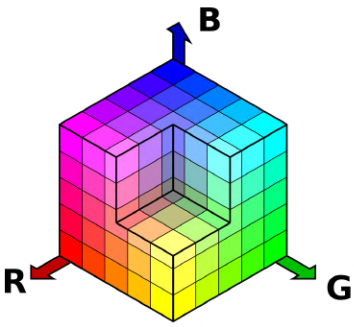
color information. A color space is effectively just a digital palette – except these colors are much more precisely organized and quantified.

As mentioned, all colors result from how the human eye or any other sensor processes light waves, but, depending on the type of media, the creation of that color comes from different methods. A color model with no associated mapping function to an absolute color space is a more or less arbitrary color system with no connection to any globally understood system of color interpretation. Adding a specific mapping function between a color model and a reference color space establishes within the reference color space a definite footprint, known as a *gamut*; in short, a gamut is the range of colors that can be reproduced with a given color reproduction system.

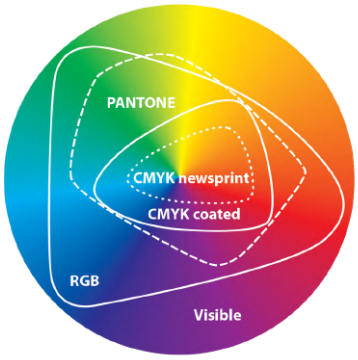
Because different devices have dissimilar color responses, which in turn differ themselves from the human senses, a color space can in theory only be defined in reference to another color space. To overcome this limit, in 1931, the International Commission on Illumination (known as CIE from French) defined an all-encompassing color space based on the average human perception, using data from experiments conducted with a small set of test subjects. Nearly a century later, this space, known as CIE 1931, remains the standard reference used to describe all other color spaces [56, 57].

Further details and in-depth, quantitative dissertations are left for consultation in the bibliographical references. Some examples of the most common color models are now briefly presented:

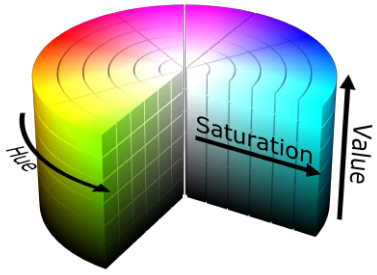
- **RGB model:** based on the additive color mixing of red, green and blue defined in a 3-dimensional Euclidean space, this is the model that naturally describes human color reception, as each of these primary colors stimulates one of the three types of the eye's color receptors without involving the other two; employed by media that transmit light – such as television or computer screens –, it will be the model on which the image description throughout this analysis is based. In the case of RGBA, a further degree of freedom (alpha) is added to quantify the transparency parameter.




- **CMYK model:** complementarily, the CMYK model is based upon the subtractive color mixing used in the printing process. The starting point is a white substrate (canvas, page, etc.), and inks are used to subtract color from white to create an image. CMYK values for cyan, magenta, yellow and black (the acronym is made considering its final letter) are stored as a quadruple. Many CMYK color spaces exist for different sets of inks or substrates; they are clearly defined by distinct mapping functions from the unique color model.



- **HSV model:** the hue, saturation and value model (HSV) is an example of a model defined in a 3D space but in cylindrical coordinates, with additive primary colors. Hue, in reality a quite faceted concept, can be grossly defined as the degree to which a stimulus can be described as similar to or different from stimuli that are defined as unique hues (red, orange, yellow, green, blue, purple); it essentially distinguishes between colors. In this model, it is an angular dimension which starts at the red primary color at 0, passing through green at 120° and blue primary at 240° and then wrapping back to red at 360°.




  The hues are organized around a central axis of neutral colors ranging from black at the bottom to white at the top. The saturation dimension resembles various tints of brightly colored paint and it describes the brilliance and intensity of a color. The value dimension resembles the mixture of those paints with varying amounts of black or white paint, representing the lightness or darkness of a color.
  The **HSL model** is quite similar to the HSV, but value (the brightness) is replaced by the lightness. The difference is that if the hue and saturation are held constant in HSV, then increasing the value increases the brightness such that a value of 1 is the brightest color with the given hue and saturation, while in HSL what happens is similar, except that all triples with lightness 1 correspond to pure white.

- **the YCbCr model:** this model is widely used for digital video. In this format, luminance (a measure to describe the perceived brightness of a color) is stored as a single component (Y) and chrominance information is stored as two color-difference components (Cb and Cr). Cb and Cr represent the difference between a reference value and the blue or red component, respectively. Among the ones seen, this is the only model fully defined by features which describe color properties instead of representing colors themselves.

The conversion between one color space and another can be performed rigorously but the process may not be trivial: due to gamut limitations, the transformation between certain color spaces could cause inexact rendering of some colors and, especially in the digital domain, rounding errors or approximations. Given that color models are defined in such different manners and live in extremely dissimilar topologies, it is often the case that a transformation between two color spaces is a meaningless concept altogether. Anyways, this problem will not be encountered in this analysis, where the RGB color model will be employed in all steps[2].

### 3.1.3 Digitalization of colors and RGB histograms

Colors can be represented by digital bits through digitization, a general term to indicate a process that converts analog information into a digital format. Digitization happens in two stages: *discretization* of the analog signal, which is continuous and hence needs to be sampled at regular intervals (the digitization frequency); and *quantization* of the samples, which requires them to be rounded so that the continuous range of analog information can be properly mapped by a finite set of discrete symbols or integer values so that they can be received by the digital acquisition system. It is obvious that the process of digitization inevitably entails some sort of approximation of the original source, with consequent loss of information. The gain, on the

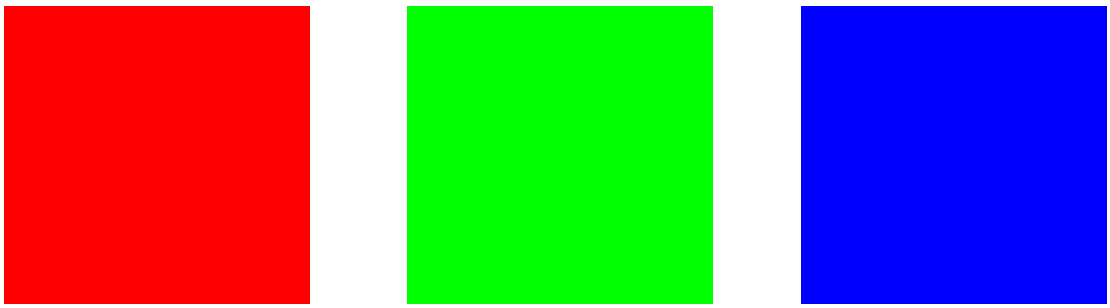
[2]For the sake of precision, the `OpenCV` package, an advanced computational tool for image manipulation which will be presented in Sec. 3.2.4, is based on the BGR formalism, which however is merely a translation of the blue and red components. This aspect should be accounted for to avoid minor programming inconveniences rather than for some solid theoretical rationale.

other hand, is that digitized information can be stored, transmitted and processed by digital devices.

It is a general fact that an image can be described by a bi-dimensional function $f(x, y)$ of light intensity in the spatial plane coordinates $(x, y)$. Of course, in general an image contains colors to it, so in fact $f = (f_R, f_G, f_B)$ or similarly for a different color space. Operationally, each image is hence described as a 3D grid of pixels: each is identified on a planar lattice whose granularity characterizes the image resolution using two spatial coordinates, while the third one more properly refers to the color channels. This means that a single pixel actually is a triplet of numbers and these three numbers represent the amount of red, green, and blue in the color assigned to that given unity of image.

The RGB color model is implemented in different ways, depending on the capabilities of the system used. By far the most common general-used format is the 24-bit implementation, with 8 bits for each primary color, corresponding to $2^8 = 256$ discrete levels of color per channel, known as *pixel intensities*. Any color space based on a 24-bit RGB model is thus limited to a range of $256 \times 256 \times 256 \approx 16.7$ million colors. This is not the only possibility, though: some implementations use 16 bits per component for a total of 48 bits, resulting in the same gamut – which, as seen, it is a quantity defined intrinsically by the color model – but with a larger number of distinct colors (with as many as $2^{16}$ discrete levels per channel, approximately 281.5 trillion colors can be discriminated). The same principle applies for any color space based on the same color model, but implemented in different bit depths; 24 bits are often considered sufficient to represent colors with enough shades to faithfully reproduce a photograph (called *photo-realistic*).

In the chosen 24-bit format, each color is conveniently represented by 8 bits, so a total of 3 bytes is required to fully identify a pixel.



**(A)** $(R, G, B) = (255, 0, 0)$
*Binary:* 11111111000000000000000

**(B)** $(R, G, B) = (0, 255, 0)$
*Binary:* 00000000111111110000000

**(C)** $(R, G, B) = (0, 0, 255)$
*Binary:* 00000000000000001111111

**Figure 3.3:** *Digitization of the three primary colors in the RGB model both as an RGB triplet and in terms of their explicit binary representations.*

The first 8 bits are for red, the second 8 bits are for green, and the final 8 bits for blue and this accounts for how the data is structured. The numerical value of these pixels is what eventually makes an image what it is; this number somewhat rates how much of each of the three colors is present by giving it a number from 0 (meaning none) to 255 (meaning full saturation of that color)[3], because these are how many discrete levels are available for description. Fig. 3.3 should make this formalism extremely clear: the pure red in A has an R value of 255,

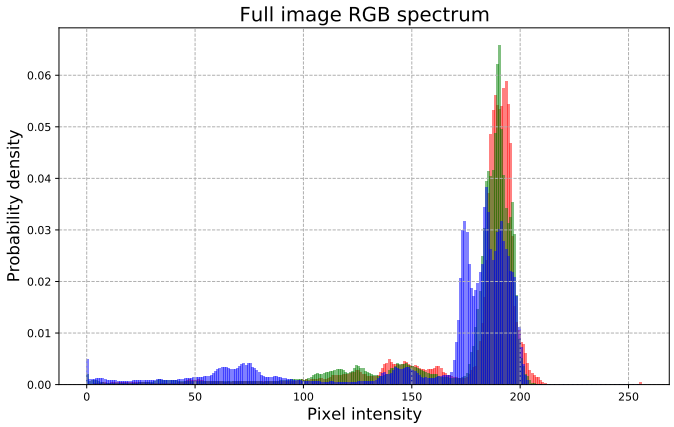
[3]In computational contexts, it is customary to enumerate items starting from 0, as this is the convention most programming languages employ. This is the reason why an 8-bit discretization is said to have 256 intensity values but their range is fixed in the interval [0,255]. While it is prominently a formal matter, it is decided to stress this aspect as yet another difference when translating from a purely theoretical to a computational framework.

represented in binary as 11111111, and of 0 for the other two. Any given pixel of this color can be represented with the triplet $(R, G, B) = (255, 0, 0)$ or in its binary form, which explicitly shows all 8 bits for each channel. The same applies in (B) and (C), since they are all pure colors; shades or combinations will of course give rise to variegated bit configurations to display the amount of each primary color in them.

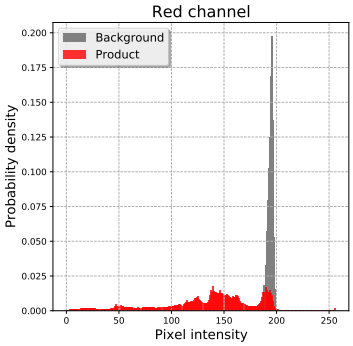
The image as it is perceived by the eye is hence recreated by pixels arranged in a precise manner; however, color histograms, also referred to as RGB histograms, are what in effect describe the image itself as far as its color distributions are concerned.
An RGB histograms shows the discrete pixel intensities and their population. It is often useful to consider a color histogram per channel since it better shows the peaks for a specific channel that may be dominant, but other concepts can easily be defined (overall histogram, average image histogram) for example when studying possible strategies to deal with noise.

For the sake of completeness, it should be noted right away that all histograms will always be normalized. Explicitly, this is done dividing the counts in each bin with the total number of counts. The goal is to allow for direct histogram comparisons on what becomes a relative scale – for instance, to compare the background peaks between two images, or to see the differences in how dominant the signal peaks are across different products.
Moreover, the histogram bin size will be kept constant at 1, especially considering the fact that pixel intensity values are natural numbers.

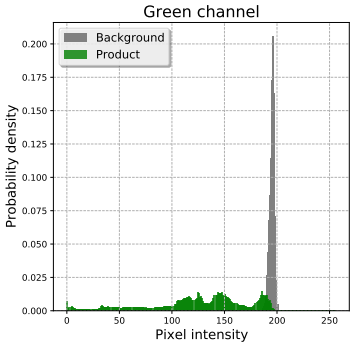

(A) *Sample image of a Ferrero Rocher.*



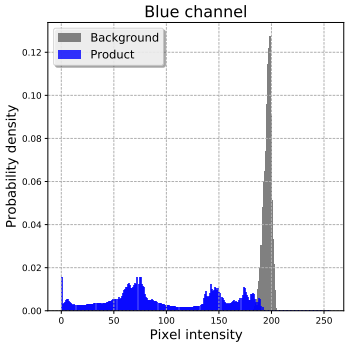

(B) *RGB histogram of the product showing the distribution of pixel intensities of the three color channels. This product is characterized by a golden packaging which causes the distributions to be multi-peaked and generally more structured.*




(C) *Red channel distribution of the product and the background.*




(D) *Green channel distribution of the product and the background.*

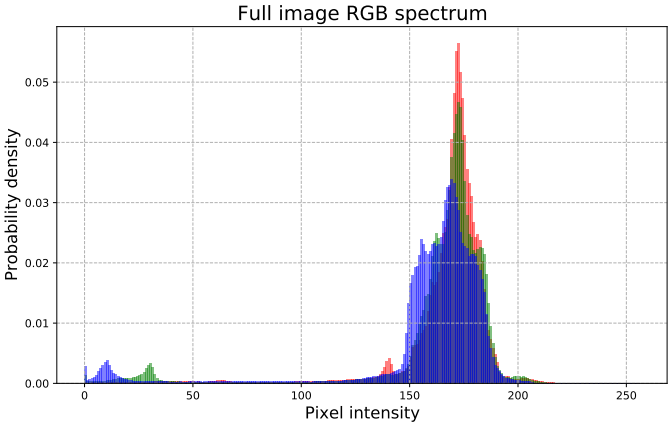



(E) *Blue channel distribution of the product and the background.*

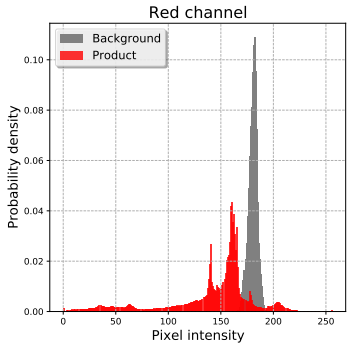
**Figure 3.4:** *A second example of a comprehensive description of another product image in terms of RGB histograms. The generally more structured product distribution tends to be dominated by the background.*

Examples for two images are displayed in Fig. 3.5 and Fig. 3.4, where per-channel histograms are shown in connection to the dominant hue for that product – which will be that real basis in the present color-based analysis. Also, the per-channel distributions are displayed all together to convey the sense of a full image representation. A first introduction to the concept of background and signal is also given, in a summarized display of the respective distributions within the considered samples. Finally, it should be noted that two products with strongly different palettes were chosen (red is dominant in Kinder Bueno, a golden tone – which happens to be a combination of primary colors in RGB – is the trademark for Ferrero Rocher), so that is it possible to compare the difference in the signals against which the algorithm to be developed should be conditioned as robustly as possible.

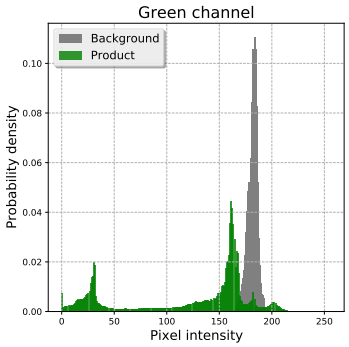

(A) *Sample image of a Kinder Bueno.*



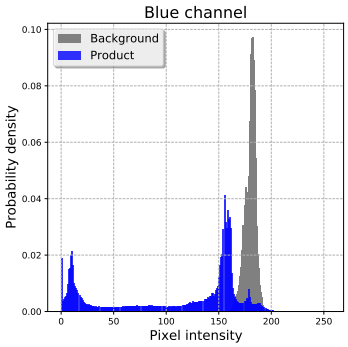

(B) *RGB histogram of the represented product showing the distribution of pixel intensities of the three color channels. The histograms are all normalized to allow for comparisons.*




(C) *Red channel distribution of the product and the white-ish background, which shows a localized and well-peaked structure.*




(D) *Green channel distribution of the product and the background.*



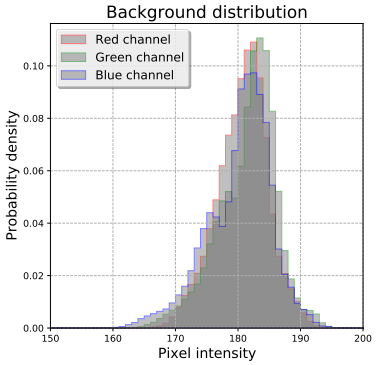

(E) *Blue channel distribution of the product and the background.*

**Figure 3.5:** *Comprehensive description of a product image in terms of RGB histograms, both globally for all channels and differentiating one by one between background and product of interest.*

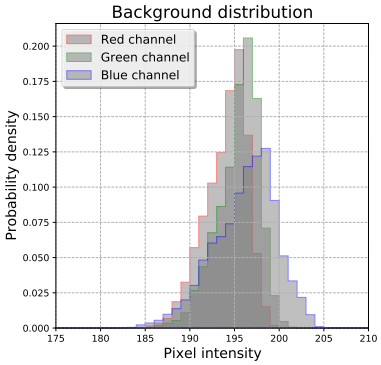
## 3.2 Background analysis and equalization

The computational part of the analysis will be fully carried out in Python, a powerful and fast high-level language which also offers several packages to deal with images. The two fundamental ones employed in these studies are `Pillow`, a general image processing tool [58], and `OpenCV`, designed for computer vision applications and containing a wide variety of sophisticated image manipulation algorithms [59].

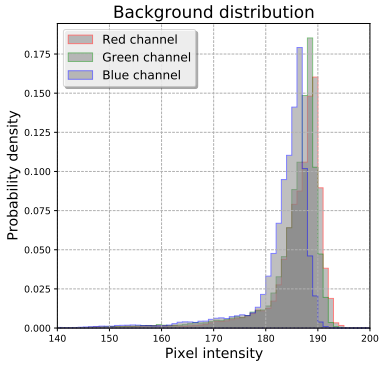
The underlying idea to approach this image analysis is to separate the background from the target (the product to be classified), as this can be configured as a starting point to define the product in the first place and then to describe it quantitatively. A focus on the distribution of the background is therefore considered as the natural beginning step for the study.



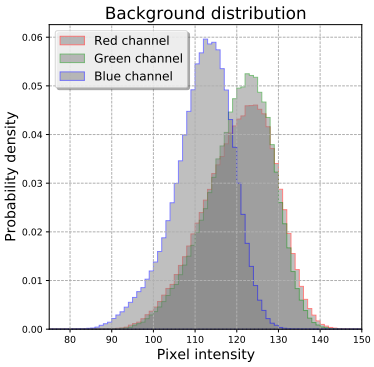

(A) *Background distribution in a Kinder Bueno image.*



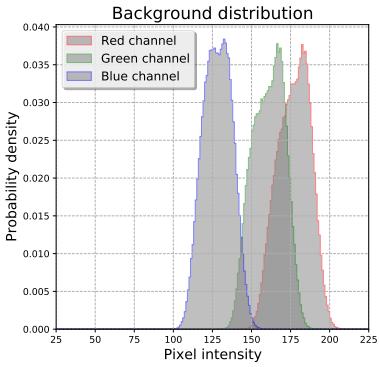

(B) *Background distribution in a Ferrero Rocher image.*




(C) *Background distribution in a Tic Tac image.*




(D) *Background distribution in a Bauli image (showing a much wider and different range of pixel intensities swept).*




(E) *Background distribution in a Barilla image (showing a marked difference in the background light conditions).*

**Figure 3.6:** *Per-channel distributions of the background of some randomly selected images from the various product classes.*

It can be noted that the background is distributed in a rather well-defined region of pixel intensities for each channel; usually, that is in the range $[160, 190]$, for instance in 3.6a and 3.6c. When this is not the case – which tends to happen especially for Bauli and Barilla as these photos were not taken in a professional environment – the peaks may be shifted, as it is the case in 3.6d. At times, the distributions of the color channels are even more spread out (3.6e) but it is even possible to identify clean bell-shaped peaks for each color channel, as distant as they may be from each other in the spectrum of intensities occupied. Anyhow, this variety can be easily dealt with by performing per-channel operations, which will be the standard approach throughout this analysis.

The possibility to localize the area of the surrounding area and the fact that these distributions have somewhat of a cleanly defined shape are indeed crucial features, as they show a first delicate aspect that needs to be addressed. In the chosen color space, pure white is represented by the RGB triplet $(255, 255, 255)$. However, a general look at said distributions or directly at the product images (for example in Fig. 3.1) immediately shows how this is not the value of pretty much any of the background pixels. The background is far from being perfectly white: images display some anisotropies in the light distribution (which appears to

be mono-directional), some shadows are present as well, at times due to the different shapes of the product packaging, and some inhomogeneities can be spotted also in the texture of the background itself.

Whatever the background RGB average value may be, a methodology needs to be found so as to scale this value back to the theoretical pure white; this process is borrowed from $\gamma$-spectroscopy and it is known as calibration with known emission lines. A color calibration is hence imposed by defining a *white requirement*: white in each image background is required to be statistically compatible with the saturation of the 3 RGB channels in the 8-bit representation, namely with the triplet $(255, 255, 255)$.

A transformation $\mathcal{T}$ is going to be needed which maps the input pixels in the background regions to output pixels whose white is properly balanced, enhancing the contrast of the image. Histogram equalization is an image processing technique that adjusts the contrast of an image by using its histogram, allowing the image areas with lower contrast to gain a higher contrast. Through this adjustment, the intensities can be better distributed, as the areas of lower local contrast gain a higher one. Contrast is the degree of difference between the darker and lighter parts of a photograph, essentially capturing how wide the range of color intensities is. Contrast is what makes an object distinguishable from other objects within the same field of view.

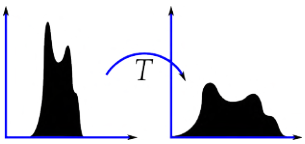



**Figure 3.7:** *A schematization of the effect that the equalization transformation $\mathcal{T}$ has on a given distribution in an histogram.*

Histogram equalization accomplishes this contrast enhancement effect by effectively spreading out the most frequent pixel intensity values or stretching out the intensity range of the image, as can be seen from the schematic representation in Fig. 3.7.
This procedure is simple and efficient, in that it is indiscriminately applied an all pixels of the image through a single calibration constant, albeit computed per channel and on the basis of the specific background of that image.

More sophisticated methods do exist, such as *adaptive histogram equalization* (ADE) or *contrast-limited adaptive histogram equalization* (CLAHE). In the former, several histograms are computed, each corresponding to a distinct section of the image, and they are used singularly to redistribute the lightness values of the image in a sort of localized equalization. It is therefore suitable for improving the local contrast and enhancing the definitions of edges in each region of an image [60, 61]. In CLAHE, a further refinement of this technique, the amplification of noise in homogeneous areas of the images is prevented with a limit on the amplification imposed by clipping the histogram at a predefined value before computing the CDF for equalization [62]. However, this methods all clash with the analysis main purpose to always find controllable and customized solutions. With the images being treated as vectors of numbers defined in a multi-dimensional color space, it makes a deeper analytic sense to prefer a solution which requires a direct and well-understood manipulation of them.

Theoretically, for each image a scaling factor $k_c$ needs to be found, defined for each color channel $c = R, G, B$, that scales the color histograms of the image to match the imposed *white requirement*. Because a calibration corresponds to an overall rescaling of the colors to match the white requirement, this can be seen to be

$$k_c = \frac{255}{\mu_c} \qquad c = R, G, B \tag{3.1}$$

where 255 is by definition (it has no error associated to it) the value of pure white and $\mu_c$ is taken to be the per-channel color average of the portion of image that needs to be equalized. This represents the background average value, which needs to be computed from a portion of the image clean from other disturbances; in this context, it is chosen as a $500 \times 500$–pixel square

at the bottom left of the image. Its average is computed as an RGB triplet represented by $\mu_c$ with its due error.

The scaling factors for each color channel in each product subset are found with their errors in Table 3.1.

| Color \ Product | **Barilla** | **Bauli** | **Bueno** | **Rocher** | **Tic Tac** |
|---|---|---|---|---|---|
| **Red channel** $k_R$ | $1.5 \pm 0.1$ | $1.6 \pm 0.4$ | $1.42 \pm 0.07$ | $1.38 \pm 0.05$ | $1.42 \pm 0.03$ |
| **Green channel** $k_G$ | $1.6 \pm 0.2$ | $1.7 \pm 0.4$ | $1.42 \pm 0.07$ | $1.37 \pm 0.06$ | $1.43 \pm 0.04$ |
| **Blue channel** $k_B$ | $1.9 \pm 0.3$ | $1.9 \pm 0.4$ | $1.44 \pm 0.07$ | $1.36 \pm 0.06$ | $1.45 \pm 0.04$ |

**Table 3.1:** *Average per-channel equalization factors obtained from a 10-image sample for each product. Errors are computed as standard deviation of the individual estimates.*

These average values are obtained by randomly selecting 10 images for each product and imposing the white requirement on their background patches defined as above. It can be noticed that two datasets have an error one order of magnitude higher ($\sim 10^{-1}$) compared to the others ($\sim 10^{-2}$); also, the absolute values of the parameters tend to be further apart between color channels in these subsets. This could be a hint of a more dishomogeneous background in these sets of images: the phenomenon is quantifiable even throughout the equalization procedure, whose main purpose is in fact to smooth out this effect and calibrate the "whiteness" in the images of the toy model under scrutiny.

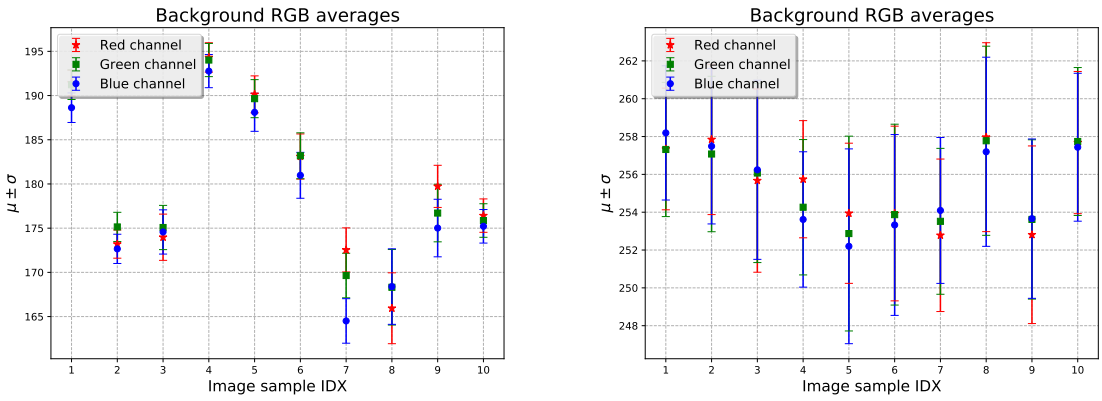
Because of the above considerations and of the variations that do occur even within the simplified model being used, during the analysis process it is preferred to equalize each image with its individual equalization factor, without resorting to average quantities. This step adds little computational burden (also because the datasets are limited in size) but it grants an ad-hoc calibration for the first crucial step of pre-preprocessing.
The scaling factor vector $\vec{k} = (k_R, k_G, k_B)$ hence needs to be applied to an image channel by channel, in a sort of tensor-vector operation exemplified in

$$\left( \begin{pmatrix} p^{(R)}_{1,1} & \cdots & p^{(R)}_{1,n_2} \\ \vdots & \ddots & \vdots \\ p^{(R)}_{n_1,1} & \cdots & p^{(R)}_{n_1,n_2} \end{pmatrix}, \begin{pmatrix} p^{(G)}_{1,1} & \cdots & p^{(G)}_{1,n_2} \\ \vdots & \ddots & \vdots \\ p^{(G)}_{n_1,1} & \cdots & p^{(G)}_{n_1,n_2} \end{pmatrix}, \begin{pmatrix} p^{(B)}_{1,1} & \cdots & p^{(B)}_{1,n_2} \\ \vdots & \ddots & \vdots \\ p^{(B)}_{n_1,1} & \cdots & p^{(B)}_{n_1,n_2} \end{pmatrix} \right) \cdot \begin{pmatrix} k_R \\ k_G \\ k_B \end{pmatrix} \tag{3.2}$$

where the scalar product means each component of $\vec{k}$ multiplies all the elements of the respective color-grid matrix (corresponding to the spatial pixels of the image $n_1 \times n_2$ in size).

To verify the fulfillment of the white requirement, the color average of a different background region from the previous one is evaluated, within each sample, before and after the image has undergone equalization, in order to check if the new average value is in fact statistically compatible with pure white.

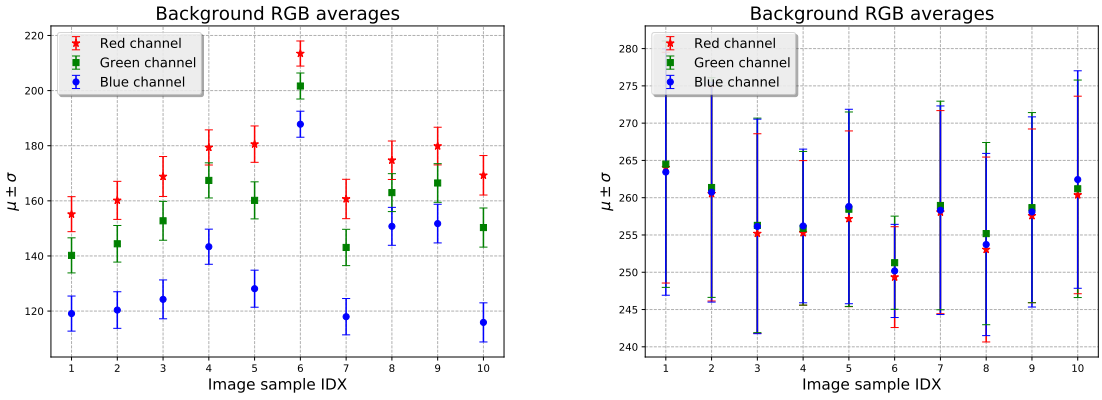
The results displayed in Fig. 3.8 show that the equalization procedure works well and it manages to transform all the background regions in order for them to be in accordance with the white requirement. The error bars appear wider in the post-equalization white averages, but the $y$-scale actually covers a narrower interval: this shows that the procedure effectively manages to reduce the variations in the background RGB averages after calibration, making it more homogeneous.




(A) *Original images.* (B) *Equalized images.*

**Figure 3.8:** *Color averages of a different background patch chosen in the Bueno subsample. Errors are the standard deviations of the values of the individual pixels in that region.*

Good outcomes are generally seen across all datasets, but with some differences. In Fig. 3.9, for instance, it is clear that there are significant differences in the background color spectrum within each image, due to the less homogeneous conditions of the setting. In (A), the color averages are sharply separated, suggesting completely different distributions from the example above (this was already pointed out when explicitly considering some background distributions in Fig. 3.6 partially hinder the equalization procedure at least in some images (index 1, 2 and 10 in particular) and in general the standard deviations of these averages are higher. However, it is worth stressing that all the average values are indeed statistically compatible with the pure white hue $(255, 255, 255)$, once again with a marked difference in the range spanned by the vertical axis in the two plots.




(A) *Original images.* (B) *Equalized images.*

**Figure 3.9:** *Color averages of a different background patch chosen in the Barilla subsample. In this case, the white requirement is respected more loosely due to the difference in the background structure within each photo.*

While the analysis step performed in this section could appear trivial from a computational point of view, it actually hides a deep conceptual leap. From the application of Eq. 3.1 onward, the objects that are being employed are not theoretically images any longer. In fact, an equalized image is not properly defined in a conventional RGB space discretized in 24 bit: some of its pixel intensity values will necessarily be outside of the domain of existence $[0, 255]$

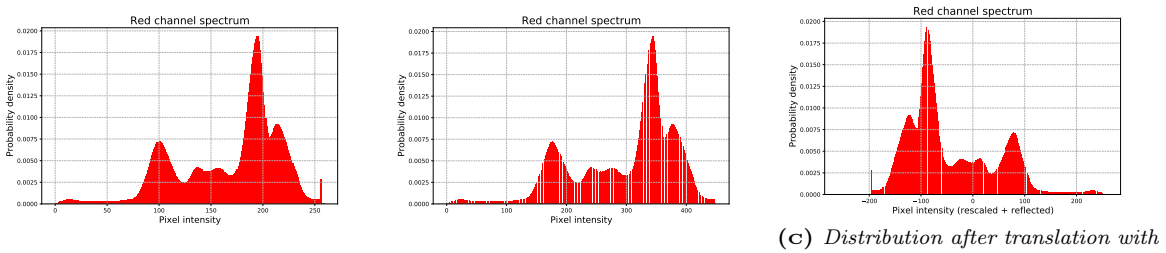
after the calibration procedure, which is obvious considering that the $k_c$ values in Table 3.1 are all above unity.

However, these objects still have the mathematical structure of multidimensional vectors, on which operations can be performed; the distribution of their values can also be visualized employing RGB histograms.
In essence, the analysis is now moving from the domain of images to the domain of histograms.

As far as visualization matters are concerned, an equalized image should be pictured as a "washed out" version of itself, with brighter whites and blurred out contours. More interestingly, its RGB histogram will clearly be stretched in range and, when rescaled with respect to the background color average (which is done for example to have the background color peaks centered in 0), negative values might end up being dominant. To avoid this, the transformation

$$\vec{p} \to \vec{p}\,' = -(\vec{p} - \vec{p}^{\,\star}) \tag{3.3}$$

is applied to each pixel $\vec{p}$ of the image, taken in its vectorial form to stress that $\vec{p} = (p_R, p_G, p_B)$ for each color channel; the quantity $\vec{p}^{\,\star}$ is the reference upon which the shift is based: it can of course be the pure white triplet $(255, 255, 255)$ or the already cited background average.
In 3.10, $\vec{p}^{\,\star} = (255, 255, 255)$.




(A) *Original distribution.* (B) *Distribution after equalization.* (C) *Distribution after translation with respect to pure white and reflection.*

**Figure 3.10:** *Example of the transformations that the red channel distribution in a random Bauli image undergoes after the equalization in Eq. 3.1 and after the translation and reflection procedure in Eq. 3.3. It is particularly interesting to notice how equalization expands the x-axis scale but it preserves the shape of the histogram.*

## 3.3 The cropping algorithm

Equalized in this fashion, images have a wider distribution especially in the region of the background, which also finds itself to be more homogeneous. This is the starting point to create an algorithm capable of separating the product (the signal to be extracted) from the background, isolating the region of interest so that the quality of the classification model can be soundly improved and cleansed to a great extent from possible spurious effects. This operation is known as *cropping* of the image, that is the act of improving its informative content by removing the unnecessary parts.

To achieve this goal, the desired approach is once again to take full advantage of the mathematical structure of images and hence to directly operate on them. To exploit color as a quantitative variable for this goal, it has already been noted how the background can be found in a fairly recognizable region of pixel intensities; once the images have been equalized, this effect will be even more obvious.

It hence makes sense to define a background-reference value taken to be the average of a patch of image which represents well the background conditions. In this analysis, this was chosen to be the RGB average of a $500 \times 500$–pixel square at the bottom left of each image, in accordance with the previous approach.

This background average triplet can be subtracted from the image itself. If a logical mask is then introduced according to

$$\texttt{|image - background| < threshold} \tag{3.4}$$

which could be employed to filter out unwanted pixels and retain only the ones classified to be product. This will be a key-step in the definition of the cropping strategy. Of course, a quantitative parameter will need to be introduced in order to control how this filtering mask gets created in order to actually *filter* the image.
Such a variable will be a key descriptor not only in terms of a justifiable quantitative analysis, but also to efficiently tune the algorithm for best performances and evaluate the results obtained.

In the choice of how to actually perform the slicing of the images to retrieve their products, the peculiarity of the dataset is once again taken advantage of. Considering that in this simplified scenario all the samples only have one product and that said products are all grocery items with a regular shape (this analysis is eminently color-based), it seems reasonable to opt for a cropping solution involving the creation of a box that defines the perimeter to be retained. This solution is extremely intuitive to implement and it grants impressive results, but it is clearly weakened by the geometrical constraint: products that do not have a parallelepipedal shape (which is not the case, at least within the chosen toy model framework) or that are found at an angle are likely to undergo an inaccurate cropping, which will necessarily leave portion of the background in it.


(A) *Two examples of a well-performed cropping.* (B) *Two examples of a poorly-performed cropping.*

**Figure 3.11**

### 3.3.1 Threshold analysis

With the basic principles of the cropping algorithm being outlined, a study needs to be carried out to quantify the `threshold` parameter. This is in fact the only quantity that regulates the behaviour of this cropping procedure and it needs to be well understood so that its effects on the quality of cropping can be measured and an ideal value for it can hopefully be identified and fixed throughout the analysis that follows.

This analysis hence ought to start from understanding the effects of an increasing threshold on the cropping to be performed on the various images. To do so, a sub-sample with 10 images of Kinder Bueno is selected. A sort of true value of cropping can be defined by operating a manual cropping on to each image – which is in itself a strategy far from being immune from errors, especially considering the difficulty in keeping a pixel precision from a screen when cropping images that indeed have a limited noise, but that still show various products at different angles and zooms. This true value is therefore taken as the ratio between the area (in arbitrary units that can be thought of as the number of pixels enclosed in the region) of the cropped portion of the image and the whole image itself; a similar line of thought is chosen in the calculation of the percentage cropped automatically by the algorithm.

A parameter should be defined that allows comparisons between the percentage cropped by the algorithm and the one taken as reference; however, in order for different images to be compared, this parameter should be normalized to the reference value at any given threshold. A good synthesis of the aforementioned needs is found in the following definition of $\Delta_{\%}$ to quantify the quality of cropping as function of the input `threshold`,

$$\Delta_{\%} = \left| \text{mean} \left( \frac{\%_{alg} - \%_{true}}{\%_{true}} \right) \right| \tag{3.5}$$

where $\%_{alg}$ is the ratio between the area cropped by the algorithm and the whole image, $\%_{true}$ is the same ratio but with respect to the true value and the mean is taken on the considered 10-image sample to smooth out errors and possible glitches. The absolute value is employed to make this parameter even more intuitive: in case of over-cropping (the image is cropped too much and as a result only a portion of the product is actually visible), without it $\Delta_{\%}$ would in fact become negative, signaling that $\%_{alg} < \%_{true}$ and the target value has been surpassed. However, the $|\cdot|$ is useful for the problem to be formulated as a minimization task and also for visualization purposes: with any given trend, one has only to spot the minimum value to identify a good estimate of the ideal `threshold` for cropping. In order for the difference between predicted and true value to get smaller and smaller, this minimum value should be as close to 0 as possible.

Clearly, because $\%_{alg}$ is the average cropping ratio obtained through the customized algorithm, then it is a function of the `threshold` parameter, which immediately gives

$$\%_{alg} = \%_{alg}(\texttt{threshold}) \Rightarrow \Delta_{\%} = \Delta_{\%}(\texttt{threshold}) \tag{3.6}$$

exactly as wanted.
This is the graphical result obtained.

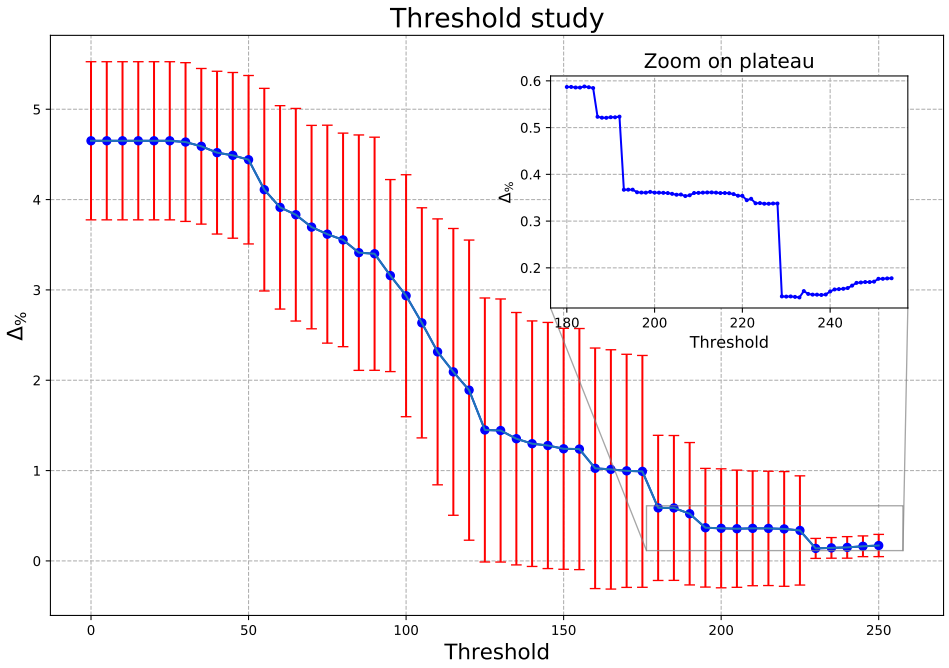



**Figure 3.12:** *Average trend of the defined parameter $\Delta_{\%}$ to evaluate the cropping quality as function of increasing values of the threshold modulating the cropping.*

The plot shows the expected trend: as `threshold` increases, $\Delta_{\%}$ approaches 0, meaning an improvement in the cropping quality, only to slightly grow back up for values above 230, likely highlighting the presence of over-cropping. The `threshold` value has an intrinsic upper bound due to the way the cropping algorithm was built. This is in fact the parameter that defines the mask from which pixels of interest are located and retained; this mask is obtained via a difference between the image pixels and the background average and if the threshold is such that no pixel satisfies the condition, the mask itself will be trivial, the coordinates ill-defined and conclusively the cropping algorithm will fail. It should be noted that the maximum value in this case reaches 250, but it might not be the case for all images[4].

The error on each estimate is computed as the standard deviation on the sample. The error bars are indeed worthy of some concern, especially considering that they are almost in all cases around $\pm 25\%$ of their relative values. This can be in part explained by the intrinsic inaccuracy of the true value computation, which requires images to be manually cropped with a coarse procedure that is also very hard to replicate in a perfectly faithful manner. Also, it should be heeded that this kind of error is made on every single side of the square defining the cropping box, so it increases quadratically when considering the area of said box. An error of about 4-5% appears to be less harmful and can be considered to be in the order of things.

These error bars should however not divert the attention from what appears to be a very promising result: an increase in threshold leads on average to better cropping. Also, within this downward trend, a plateau can be noticed corresponding to a `threshold` interval of $[180, 225]$. This seems to imply that the cropping algorithm reaches some sort of stability and the fact that this happens where $\Delta_{\%}$ approaches the target value of 0 appears of course as an indicator of robustness in the procedure.
An in-depth analysis will be needed to further understand this trend.

### Scan in threshold

From the plot in Fig. 3.12, average results are obtained (albeit from a limited sample) and a first idea of the overall `threshold` behavior is outlined, but a more comprehensive picture should be conveyed so as to perform a more informed choice for the best value for the parameter but also to understand the inner workings of the algorithm on the images, beyond the definition of descriptive theoretical quantities.

To achieve this goal, the attention is focused on a single image (Fig. 3.13A) of the sample above – one that shows few signs of dishomogeneities in the background and displays the product in a zenithal orientation to avoid spurious effects due to the box approximation – and its RGB histogram is obtained (Fig. 3.13B); here, the peaks corresponding to background and product (the latter ones clearly much smaller in scale, as the surrounding region is dominant) are pinpointed for each color channel.

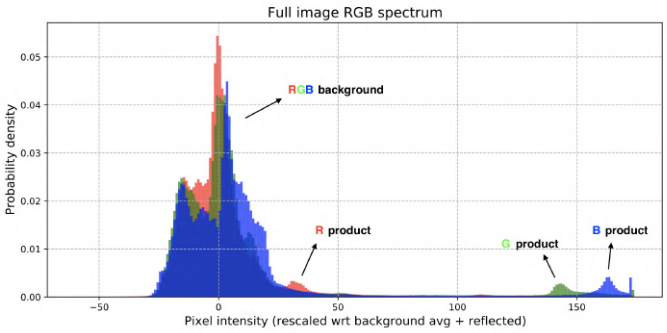
These RGB histograms, clearly normalized to allow for direct comparisons, were all shifted according to Eq. 3.3. This time, $\vec{p}^{\,\star}$ is chosen to be the background color average (computed as an RGB triplet in the same way as above), in order for the background peaks to be centered around 0.

A scan in threshold will then be performed, showing how the structure and relative height of these peaks is affected with respect to the cropped portion of the image, at different values of `threshold`. In particular, the interval where the plateau is present will be scanned in a finer way to retrieve a proof of a stable behaviour in the procedure. Not only will the RGB histogram be compared with the cropped portion of the product; the value of $\Delta_{\%}$ will be always kept as quantitative reference[5].

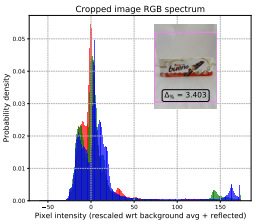
[4] Actually, it is very likely that the error bars for data points at `threshold`$> 225$ are smaller because the cropping procedure has only worked on a fraction of the total sample, hence reducing the dispersion of the values at disposal to compute the averages.

[5] It should be noted that the images displayed are the original ones, but the cropping box was determined

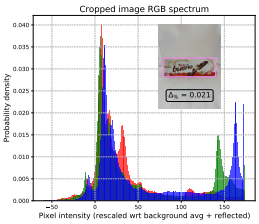

(A) *Image of a Kinder Bueno used in the threshold scan analysis.*



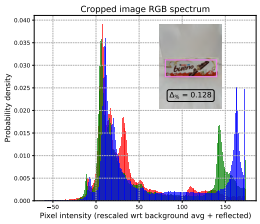

(B) *RGB histogram of the image, in which the background peaks and the signal peaks for the three color channels are highlighted. The histogram is shifted with respect to the background average and reflected to avoid the predominance of negative values.*



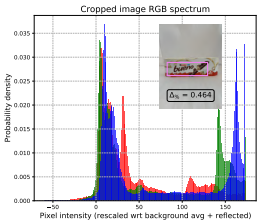

(C) *Threshold=100*




(D) *Threshold=200*




(E) *Threshold=220*




(F) *Threshold=230*

**Figure 3.13:** *Evolution of the RGB histogram distributions and the parameter $\Delta_{\%}$ at various threshold values for the displayed Bueno image.*

The scan in threshold displayed in 3.13 is able to convey extremely insightful results. It is possible to see that for `threshold=100` (C), the image is barely cropped, which results in the background peaks still being dominant and in a high value of the cropping parameter $\Delta_{\%}$.
When the plateau observed in 3.12 is entered (D-E), the expected stability is in fact observed: the cropping box fits the product very well, with a sharp decrease in $\Delta_{\%}$ (even if a slight imprecision at `threshold=220` is efficiently captured by a 5x increase in this parameter compared to the ideal scenario in D). What is interesting in these two instances is that the peaks corresponding to the product are now prevailing; also, the structures seen around 0 are not exactly centered around this value, which suggests they actually correspond to the different shadow of white of the Bueno packaging rather than to the background, which is now almost completely eliminated.
When the plateau is surpassed, over-cropping takes place: the product image is only partially obtained and, while the RGB histogram could mislead in thinking that the Bueno peaks are dominant, the bad quality of the cropping is promptly underscored by an increase in $\Delta_{\%}$.

Further tests on other images and products corroborate what has now been observed and in general prove the robustness of this algorithm. Keeping in mind that also average results have to be accounted for, it is ultimately chosen

```
Ideal threshold = 220
```

This threshold analysis was performed only on the Bueno dataset to avoid a dispersive presentation of the results. Other products or different images do display the same pattern, at times with less clear peaks or with noisier results due to more imprecise cropping outcomes.

after the equalization, following the exact recipe detailed in the above sections. In this instance, though, the coordinates defining the clipping box are saved and employed to execute the cut on the starting image, which can therefore be directly displayed.

Anyways, the images all undergo the same treatments, with a global choice for the `threshold` parameter.
This will lead to the obtainment of pre-processed data that is now ready to be employed for higher-level analyses in the definition of an optimal product classification algorithm, which will be laid out in the following Chapter.

# Chapter 4

# Classification with similarity metrics

ULTIMATELY, at this point of the analysis, the images have undergone their due pre-processing in two stages: firstly, through an equalization procedure to calibrate the whiteness of their background, then via a cropping routine to isolate the product from the rest of the image. Still in the boundaries of the defined toy model, at this point the images should in theory only be showing the portion containing a product, with the background regions mostly expected to have been filtered out in the previous step.

In essence, a feature extraction from the input images has just been performed, using customized and controlled criteria. These features will represent the building blocks for the class reference templates, needed in the implementation of the classification algorithm that follows. This constitutes a significant difference with a deep learning model, where instead the feature extraction process is carried out automatically at training time.
It should be stressed that since the coordinates for the cropping box that defines the relevant portion of each image are obtained after each image has been equalized, the cropping itself happens on the original ones; this implies that this stage of the study goes back to employing actual images, together with the constraints they require for a proper definition in the RGB color space.

The aspect of classification now needs to be tackled. In a multiclass (or multinomial) classification, the problem is posed of assigning a class membership to instances among three or more available – a problem where only two classes are available is usually referred to as binary classification. A machine-learning algorithm for classification requires two components so that it can be well-defined:

1. a **class template** that in a way defines the class itself becoming its representative, so that other instances can be led back to it to quantify their similarity;

2. a **metric** to effectively measure the *similarity*, making it possible to retrieve a numerical value that the algorithm can employ to predict the product class.

As far as the template is concerned, a quantity needs to be devised that is fit to act as a sort of archetypal entity to be used as the second term of comparison in a similarity relationship. In this study, each class will be represented by an *average histogram* of the color histograms of all the images belonging to that class. The averages will be performed bin by bin, so the theoretical foundations of RGB histograms remain intact. This strategy will release the classification stage from the dependency on the training set size, as each image will only have to be compared to the average histogram of each product class. As it will be seen in the dedicated section below, this will prove to be a valuable advantage especially considering that this algorithm was developed to treat images, which are particularly heavy to store and computationally more challenging than plain numerical data.

Regarding the classification metrics, they are the mathematical descriptors to quantify *how similar* two images are. The similarity has to be formally defined both because a univocal

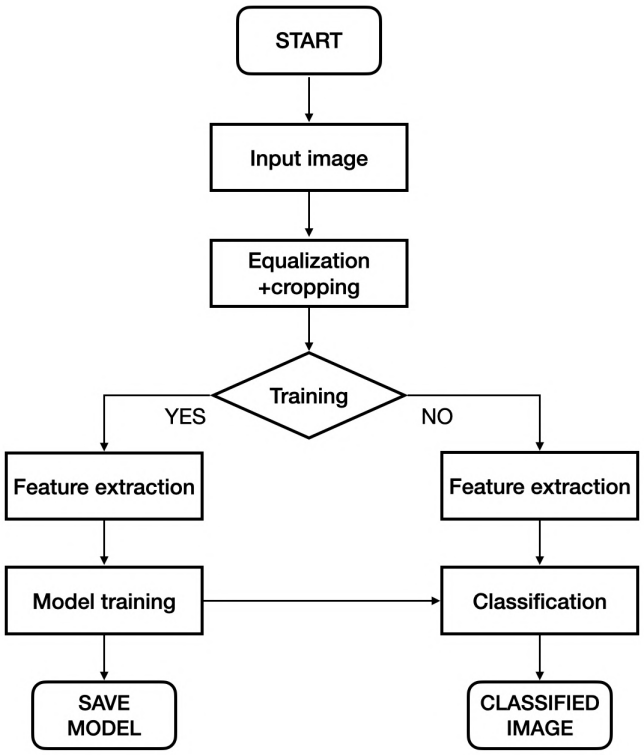



**Figure 4.1:** *Flowchart of the customized ML classification algorithm developed in this first part of analysis.*

parameter is needed for the sake of a rigorous analysis and also so that the algorithm is capable of assigning a class to each image. This operation will be carried out by finding the class for which the distance between its average histogram and the RGB histogram of the image itself is minimized, meaning that the similarity is maximal. This will happen on a per-channel basis, but a comprehensive definition of distance can be derived for instance by summing in quadrature the three values, to resemble the formal structure of the Euclidean distance.
Because this is such a crucial step in the analysis, three metrics will be employed, in the form of three statistical tests borrowed from various fields: the Kolmogorov-Smirnov test, the $\chi^2$ test and the Manhattan test (in the form of Manhattan distance minimization). Each test will provide a value that can be interpreted as a distance between distributions, hence assigning a processed image to a certain class after the comparison with the templates; the results obtained throughout the analyses will shed more light on the strengths and downsides of each metric.

---

This chapter delves into the aspect of classification of the images. The reference templates for each product class will be defined, resorting to the concept of average color histograms; shortly thereafter, the similarity between them will be formalized quantitatively by defining said similarity metrics. The results of this embryonic customized ML algorithm will be then displayed and thoroughly commented, highlighting the promising capabilities as well as the limitations under which the model was built and trained. These results will be then compared with the classification capabilities of Custom Vision AI, a state-of-the-art Microsoft service to train and evaluate computer vision models.
Lastly, an application in a real case scenario will be presented to show how an analysis can

be translated into a concrete, product-oriented application thanks to the services available at Porini within the Microsoft infrastructure.

## 4.1 Choice of a class template

The dataset is divided into 5 classes and these are the possible ways in which an unknown image can be classified. To begin with the layout of the classification part of the algorithm, a descriptor needs to be defined for each class so that it can serve as a fixed reference to embody that specific class. Of course, this is a rather delicate step: such a descriptor will be the only class-related information employed when categorizing images, so it will need to enclose a representation of that class as complete and faithful as possible; at the same, it will have to be easily comparable with said images and, in accordance with the foundational principles of this analysis, interpretable and solidly understood.

Consequently, the idea is to resort to *average RGB histograms*, computed on a per-class basis. In practice, the RGB histograms that describe the cropped images in the training sets of each class are averaged bin by bin. The result are 5 average histograms, one for each class, where for every pixel intensity value – they remain unchanged since they depend on the way images have been digitized – the corresponding counts are the average of the counts of that class images for that specific intensity value. This newly defined entity has of course no actual connection to any existing image, but it can be interpreted as the distribution mathematically representing the "average image" for each given class.

These tools have an immediate derivation and a very intuitive nature. They allow to perform not only inter-class comparisons but also intra-class ones, for instance to quantify, within a certain class, the distance of each image with its average. Moreover, their formal structure resembles in all respects that of the RGB histograms considered so far, as it can be also seen in Fig. 4.2, representing the average histogram of the Bueno class. Here, it can also be noticed that the most significant features of the Bueno distribution are preserved, such as the red peak at around intensity 40, the green and blue ones at about 140 and 160 respectively (corresponding to the trademark red of this brand) or the multiple peaks at intensities 10-20 (corresponding to the white portion of the Bueno packaging). In addition, averages rid the histogram of extemporaneous fluctuations and make it visibly smoother, which is clearly a further endorsement in this choice for class representatives.

An alternative to average histograms could have to employ a sort of clustering logic, for example with a $k$-nearest neighbor ($k$NN) classification algorithm. $k$NN is a simple ML algorithm that stores all the available cases and classifies the new data based on a similarity measure, captured by calculating the distance between points on a graph. In essence, it is used to classify a data point based on how its neighbours are classified; the parameter $k$ represents the number of nearest neighbors to be considered when classifying new data points.

However, average histograms have a series of significant advantages. First off, they are independent from the size of the training set and hence make the model much more versatile and compact storage-wise. A $k$NN algorithm, instead, is highly dependent on it, both in terms of computation costs – at each query, the distance to all the other training samples needs to be computed – and storage-wise – because the full training set is necessary to make it work and not just the averages of each class. Moreover, while bigger samples evidently lead to more robust and statistically more significant mean values, average histograms can be defined also with a fairly restricted dataset such as this one. A clustering algorithm, to be reasonably performant, needs to have data points in the order of the hundreds. Lastly, the clustering algorithm requires an estimation of the $k$ parameter, which is not done via a straightforward formula but rather with some empirical tuning; the average histograms employed only require undemanding mean computations.

The full implementation of the algorithm gives as output a JSON file where, among others,

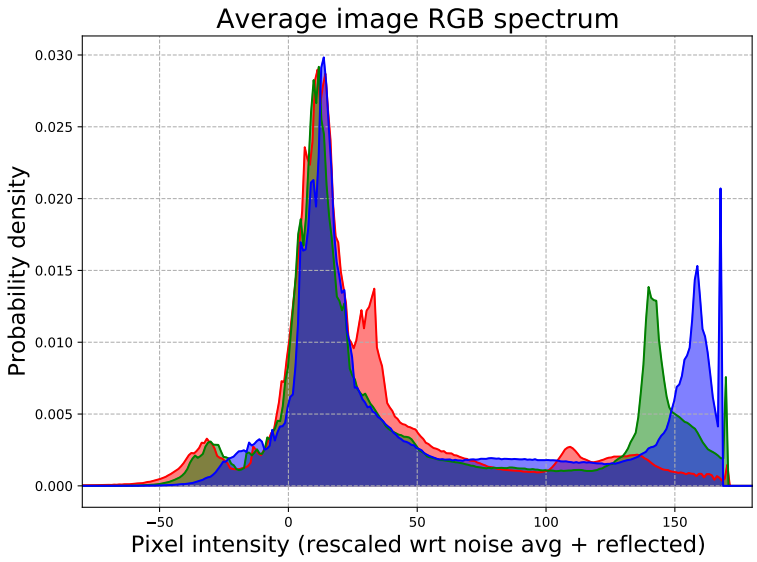



**Figure 4.2:** *Average histogram for the Bueno class, obtained by computing a bin-by-bin mean of the counts of the training set images for this class.*

these average histograms are stored. This file, which is hardly 0.5 MB in size, is the only needed item to fully characterize the classification model and to use it on a test set. This grants unprecedented versatility to the developed model: on the one side, during training only few operations aside from average computations need to be performed, which is immensely less computationally demanding than DNNs and greatly reduces performance times; on the other hand, only a limited amount of storage is required when testing it, which is a further asset considering that the model works with images, which can be a quite memory-onerous data type.

## 4.2 The similarity metrics

Having found in average RGB histograms the entities that will act as the means of comparison in the classification task, it is just as crucial to define metrics that will allow to perform the actual quantitative measurement of similarity. The simple schematic in Fig. 4.1 clearly shows how these will be the tools that make it possible for the algorithm to go from a duly pre-processed image (which means it was white-calibrated through equalization and cropped to remove the background) to a classification guess for that image. Once again, in the spirit of the analysis to proceed with justified and well-known steps, the goal is to find quantities that are both well-understood and suitable for the kind of predictions needed.

For the sake of precision, the algorithm does not measure the similarity itself, which mathematically is a rather impalpable concept. Instead, as a proxy, a distance will be measured between the distribution to be classified and the template with which it is being compared. To all extents, this distance can be thought as a metric defined in the 256-dimensional space where the distributions being tested also have their natural domain of existence; in such a high-dimensional topology, these distributions can be considered points. It seems reasonable to assume that in this space of color distributions, if a distribution finds itself at a lower distance with respect to a certain reference A rather than a different reference B, then it will be more similar to A. Concretely, the coordinates in the $(\text{pixel intensity}, \text{counts})$ space describe the structure of the distribution, which in turn is a descriptor for the starting image; if two sets of coordinates have smaller differences in their values, the image being described by the

tested distribution will have a higher resemblance to the "average image" representing the class of a given product. Comparing a given distribution with multiple class templates and finding the one located at minimal distance from it will translate into finding the class with respect to which the image to be classified has maximum similarity.

While reasonable and mathematically robust, this reasoning is completely based on the reliability of the chosen metric, which needs to be robust against negligible fluctuations but at the same time it has to pick on those differences that are fundamental to discriminate one product class from the other. A priori there is no way to decide how to actually perform this task; because this is a cornerstone step in identifying a class for an image, it was decided to introduce three different metrics: the Kolmogorov-Smirnov test, the $\chi^2$ test and the Manhattan distance. The first two are not specifically meant to be metrics, but they preserve all the topological properties of a distance: they are capable of measuring the closeness of two distributions and therefore respond well to the purpose of this analysis. Each test will be shortly detailed below; in the Results section the advantages and weaknesses of each will be made clear.

### 4.2.1 The Kolmogorov-Smirnov test

The Kolmogorov-Smirnov test (KS test) is a non-parametric statistical test to verify the degree of similarity between monodimensional probability distributions. A non-parametric test is a technique that is not based on the assumption that the data comes from a specific distribution. This makes it unnecessary to impose starting assumptions or even to introduce ad hoc parameters to model said distribution, all aspects that contribute to enlarge the robustness of this test, meaning it performs reasonably well in a wide range of cases.

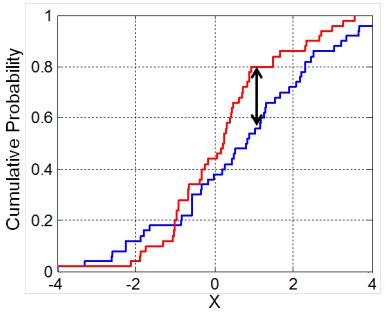



**Figure 4.3:** *Representation of the two-sample Kolmogorov–Smirnov test; the red and blue lines are the two empirical cumulative distributions being compared, while the the black arrow is the distance measured by the KS test.*

In essence, the KS test measures the distance between two samples as the maximum of the differences between the cumulative distribution functions (CDFs) of the two samples, in absolute value.

This is clearly displayed in Fig. 4.3, with the CDFs obtained as the cumulative sums of the sample distributions, which are clearly normalized.

In formulae, this value, at times referred to as KS score, is given by

$$KS = \max_k \left( \left| C_{test}^{(k)} - C_{ref}^{(k)} \right| \right) \tag{4.1}$$

where $C_{test}^{(k)}$ is the cumulative distribution of the test sample (the image to be classified) and $C_{ref}^{(k)}$ of the reference (each class template against which the classification is to performed), that is

$$C_{test}^{(k)} = \sum_{k=0}^{255} d_k^{test} \qquad C_{ref}^{(k)} = \sum_{k=0}^{255} d_k^{ref} \tag{4.2}$$

with $d_k^{test}$ and $d_k^{ref}$ the $k$-th occurrence of the distribution of the image and the reference respectively. As mentioned, this computation happens on a per-channel basis, meaning that each image will have three KS scores for each template with which it gets compared. These three values can give specific insights on how well the classification procedure operates with respect to each color channel, so at a finer level; however, it is indeed more convenient to express

the KS score as an overall distance value when a final outcome for categorization needs to be produced.

The two-sample KS test is one of the most useful and general non-parametric methods for comparing two samples. It is sensitive to differences in both location and shape of the empirical cumulative distribution functions of the two samples, even if it tends to be more sensitive near the center of the distribution.
It can be employed either to decide if a sample comes from a population with a specific reference distribution (the so-called one-sample KS test) or to compare two samples and determine if they come from the same underlying distribution (two-sample KS test). The second kind of test is the one that will find application in this study, where empirical distributions are always at disposal and it would appear far-fetched to impose known distributional forms on them. In this context, the KS test will only be used as a metric to quantify distances, so details about the Kolmogorov-Smirnov statistics and in-depth considerations about its null hypothesis and related concepts are disregarded in this study.

### 4.2.2 The $\chi^2$ test

The $\chi^2$ is among the most renowned statistical test to measure how a model compares to actual observed data. The chi-square statistic does so by comparing the size of discrepancies between the expected results and the actual results; for it to be applied, the standard deviations o. Also in this case there are several deep statistical considerations beneath, which are however overlooked in favor of a definition of this test rather as a metric $\chi^2 = \chi^2(O, E)$ which quantifies the distance between the observed distribution $O$, which as usual is the image to be classified, and the distribution $E$ represented by the 5 average histograms.

The $\chi^2$ is defined as

$$\chi^2 = \sum_{k=0}^{255} \frac{(O_k - E_k)^2}{\sigma^2_{E_k}} \tag{4.3}$$

where $O_k$ and $E_k$ are the $k$-th coordinate of the observed and expected distributions respectively, while $\sigma^2_{E_k}$ is the standard deviation on the $k$-th coordinate of the expected distribution. It should in fact be reminded that the expected distributions consist of average RGB histograms computed from the set of images belonging to each class, hence a standard deviation will be naturally associated to the bin-by-bin average.

From Eq. 4.3, it can be deduced that $\chi^2 = 0$ indicates a perfect agreement between the two distribution, as it implies $(O_k - E_k) = 0\ \forall k$, which is clearly a statistically unrealistic result; in general, $\chi^2 \lesssim N$ – where $N$ is the dimensionality of the problem (N=256 in thi case) – indicates an acceptable agreement between observed and expected data, while $\chi^2 \gg N$ signals a significant disagreement. It is again timely to stress that this quantity will be employed as a proxy to measure distance rather than to actually quantify agreement between two distributions. However, it could still be useful to introduce and work with a more intuitive quantity, the so-called reduced chi-squared, $\tilde{\chi}^2$, defined as

$$\tilde{\chi}^2 = \frac{\chi^2}{N} \tag{4.4}$$

where $N$ is the number of degrees of freedom of the problem (which, as it happens, also corresponds to the expected value of $\chi^2$). Given that histogrammed distributions are being employed, $N = N_{bins}$, number of bins the histogram is constrained to have. The advantage of $\tilde{\chi^2}$ is that, being a sort of normalized quantity, its reference value is 1 and hence it allows for comparisons on a unitary scale, which is more intuitive especially as far as the degree of accordance between the distributions is concerned.

### 4.2.3 Manhattan distance

The Manhattan distance is a metric in which the distance between two points is the sum of the absolute differences of their cartesian coordinates. Regardless of the space dimension, this metric is inherently defined on a lattice; because the 2-dimensional visualization resulting in a grid is immediate, this metric is also commonly known as taxicab geometry or the city block distance – all with references to the celebrated architectural structure of New York City.
Quantitatively, the image color distributions can be thought as points in the (pixel intensity, counts) space, which is a 256-dimensional variety in the case of 8-bit digitization. This abstraction makes it straightforward to write

$$d_{Manhattan} = \sum_{k=0}^{255} |O_k - E_k| \tag{4.5}$$

where $O_k$ is the $k$-th coordinate of the observed distribution and $E_k$ the one of the expected distribution.

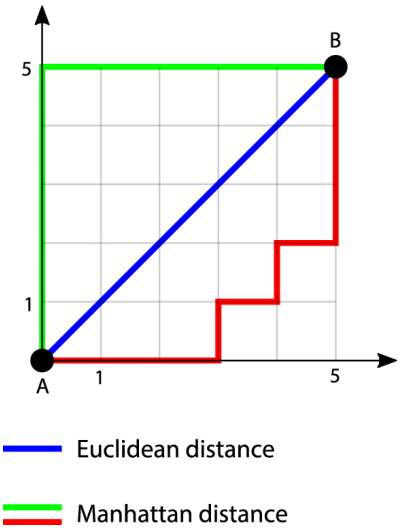



**Figure 4.3:** *Representation of Euclidean distance andManhattan distance between two points. It can be seen how in a taxicab geometry the shortest path between two point is not univoquely defined.*

Considering that the pixel intensity values are fixed in the range $[0, 255]$, it is intuitive to see that $d_{Manhattan}$ results in the plain difference between the histogram counts of the two distributions being compared. This definition also unveils that this is an $\ell_1$ metric, also labeled as absolute-value norm.

The Manhattan distance is the only metric between the chosen ones that natively accounts for the discrete nature of a digitized color space. The schematic in Fig. 4.3 also shows how with such a metric the distance between two points is not uniquely defined: multiple non-trivial paths with shortest length can be found, differently from what happens in the Euclidean case.

## 4.3 Results

### 4.3.1 Classification performances

In order to convey a bigger picture portrait of the classification capabilities of each metric, a test set is introduced containing 10 images for each of the 5 classes, for a total of 50 samples. These images have not been used in the determination of averages and are therefore unseen to the algorithm. Instead of focusing on the per-channel distances, for now a unique value is defined as

$$d = \sqrt{d_R^2 + d_G^2 + d_B^2} \tag{4.6}$$

where $d_R$, $d_G$ and $d_B$ are the per-channel values actually obtained by each metric; clearly $d = d(O, E)$ function of the observed distribution O and the expected (the average) E, because $d_c = d_c(O, E)$, $c = R, G, B$. This formula is chosen because of its resemblance with the Euclidean norm, which is one of the most natural ways to define a distance between two points in an $N$-dimensional space.

Thus, each image is compared with the 5 average histograms obtained in the training stage and the distance from each is computed according to Eq. 4.6; the class membership for that image is assigned considering which reference has minimal distance from it. This process is

repeated for each of the three chosen metrics, as they have different ways of computing the per-channel distance values and, consequently, $d$.
The results are presented in the following confusion matrices, tables to evaluate the classification performances of an algorithm by simply showing how its predictions compare to reality.

| Real \ Pred | Barilla | Bauli | Bueno | Rocher | Tic Tac |
|---|---|---|---|---|---|
| **Barilla** | 6 | 3 | 1 | 0 | 0 |
| **Bauli** | 0 | 10 | 0 | 0 | 0 |
| **Bueno** | 0 | 0 | 10 | 0 | 0 |
| **Rocher** | 0 | 0 | 0 | 10 | 0 |
| **Tic Tac** | 0 | 2 | 0 | 0 | 8 |

**Table 4.1:** *Confusion matrix obtained using the Kolmogorov-Smirnov metric.*

| Real \ Pred | Barilla | Bauli | Bueno | Rocher | Tic Tac |
|---|---|---|---|---|---|
| **Barilla** | 10 | 0 | 0 | 0 | 0 |
| **Bauli** | 0 | 10 | 0 | 0 | 0 |
| **Bueno** | 0 | 0 | 10 | 0 | 0 |
| **Rocher** | 0 | 0 | 0 | 10 | 0 |
| **Tic Tac** | 0 | 0 | 0 | 0 | 10 |

**Table 4.2:** *Confusion matrix obtained using the $\chi^2$ metric.*

| Real \ Pred | Barilla | Bauli | Bueno | Rocher | Tic Tac |
|---|---|---|---|---|---|
| **Barilla** | 10 | 0 | 0 | 0 | 0 |
| **Bauli** | 0 | 9 | 1 | 0 | 0 |
| **Bueno** | 0 | 0 | 10 | 0 | 0 |
| **Rocher** | 0 | 0 | 0 | 10 | 0 |
| **Tic Tac** | 0 | 1 | 1 | 0 | 8 |

**Table 4.3:** *Confusion matrix obtained using the Manhattan metric.*

Overall, the prediction capabilities of the algorithm appear to be good even regardless the metric. It appears that three classes are the ones more prone to errors: Barilla, Bauli and Tic Tac. In effect, the first two are the ones showing a higher degree of dishomogeneity in their background (as noted, for instance, in the computation of the equalization factors in Section 3.3), which could constitute an obstacle; the Tic Tac, instead, are characterized by a white packaging, which could cause disturbances in equalization hence lessening the effect of the pre-processing for classification.

Some differences are clearly based on the choice of the metric. The Kolmogorov-Smirnov test is indeed the one showing a higher number of misclassified items: this might be due to the fact that it only considers a single point of the two distributions being compared, the one corresponding to their maximum distance, and it is therefore less sensitive to low-level differences. The Manhattan test, which considers differences between all coordinates, already shows a marked improvement, likely due to the fact that the full distributions are considered in the distance computation, albeit in the form of a simple subtraction.
The $\chi^2$ test shows perfect scores for all classes, showing a stronger prediction power than the other metrics. Not by chance, this is the only test that accounts for Poissonian fluctuations $\sigma_E$ in the histograms, a further degree of complexity to detail the features of the distributions involved.

It should be pointed out that this result showing perfect accuracy is also linked to the size of the test set. A 50-image test set is somewhat a limited one: it indeed conveys a general sense to compare the three tests, but it does not allow for sharp conclusions. It is fairly expected

that with a bigger sample the classification accuracy of the $\chi^2$ would not be exactly 100%, but this does not alter the prediction power of this specific test.

### 4.3.2 Per-channel distances

Per-channel distances are now considered for each metric. In fact, it has been mentioned that color is a fundamental discriminating factor throughout this analysis, so a finer analysis in this respect could offer deeper hints on the inner workings of the classification process.

All the images in the test set are hence compared in turns with the 5 average histograms defined per class, employing one metric at a time. The distances are then plotted so that it is possible to see not only which is the closest image to the reference (and if it is the correct one), but also how the others are arranged. This is also done on a color channel basis, so the individual distances can be compared for each sample and class.

A focus is made on the Barilla dataset, which is the one that showed the highest criticality. The Barilla average histogram is compared with all the 10-sample test sets corresponding to each product.
As a matter of fact, in Fig. 4.4a, the Kolmogorov-Smirnov metric shows some misclassifications in the blue and especially the green channels, where the Bauli images are the ones with minimal distance to the Barilla reference in half the instances. However, the red channel distances are all correctly minimized, which acts as a counterbalance on the overall classification. No sample is in fact mistakenly classified in more than a single channel.

A similar situation is also seen in the Manhattan test in Fig. 4.4b, where again the green channel is highly confused with the Bauli product. In this case, in 3 instances the Barilla sample is not the one being minimized in the majority of the channels (two or more). It should be pointed out, though, that, when choosing other average histograms as fixed references, the Manhattan test has proven to be a more robust metric than the KS test.

There is anyway little doubt that the best classification metric is the $\tilde{\chi}^2$, as evidenced also in the previous Section. In Fig. 4.5, only a handful of instances are misclassified in single color channels, but the test shows strong prediction capabilities. It should also be noticed that the $y$-axis is in log scale, which makes the distances between points even more marked.

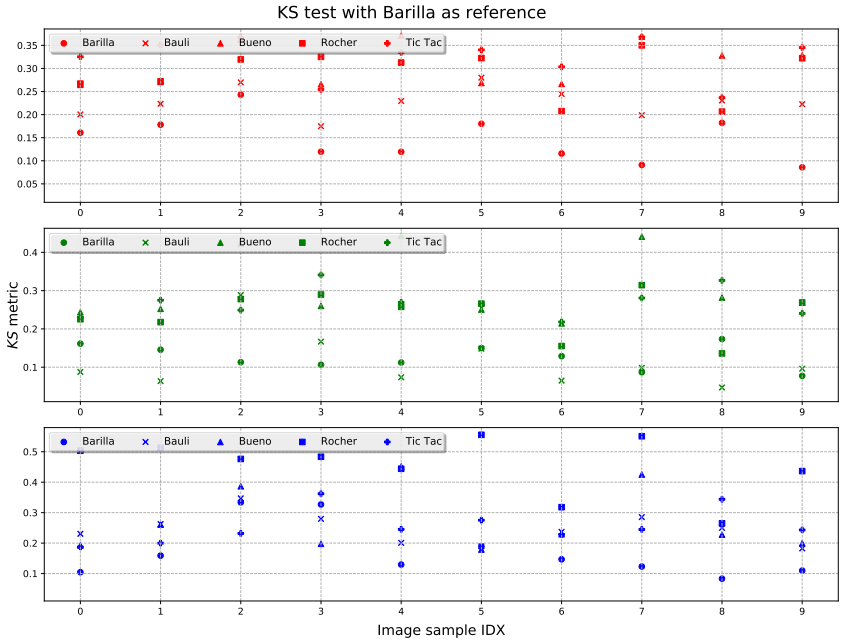
This is even more true for other datasets. For example, in Fig. 4.6, the Bueno average histogram is taken as reference. In this case, all the instances in all the channels have a correctly minimized distance with the reference, also in accordance with the previous results.

The $\chi^2$ metric definitely qualifies as the best performing one. It seems reasonable to assume that the fact that the standard deviations of the average histogram used as references are accounted for in the calculation proves to be a game-changer. In fact, the test appears to be more robust to statistical fluctuations, which could indeed occur in an average-defined classification model.

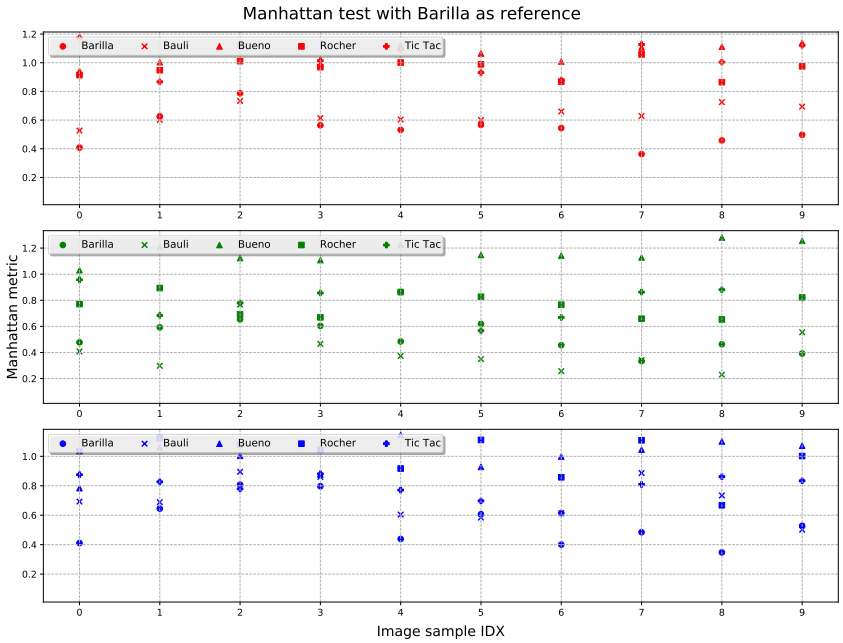
## 4.4 Comparison with Custom Vision AI

The Azure Custom Vision AI is a Microsoft service dedicated to image recognition. It makes it possible to build, deploy and refine tailor-made image identifiers, taking advantage of an outstandingly intuitive graphical interface. An image identifier is a ML algorithm that applies labels represent classes to images, according to their visual characteristics.

In practice, images are submitted and manually labeled; the rest of the process is fully automated, with the online service managing data training and accuracy computations. The end-user is then free to perform its own classification tests on the model or to modify its training. The platform offers several varieties of algorithms that are optimized specifically to recognize images with certain subject material, among which one designed for retail products, the main object of this analysis.




(A) *Per-channel distances computed with the Kolmogorov-Smirnov metric for each product 10-image sample using the Barilla average histogram as reference.*




(B) *Per-channel distances computed with the Manhattan metric for each product 10-image sample using the Barilla average histogram as reference.*

**Figure 4.4**

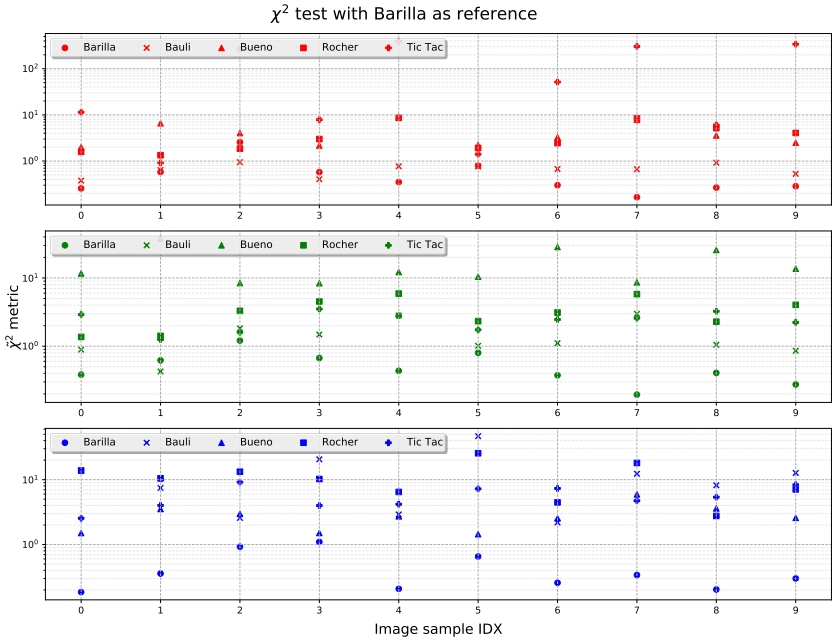



**Figure 4.5:** *Per-channel distances computed with the $\chi^2$ metric for each product 10-image sample using the Barilla average histogram as reference.*

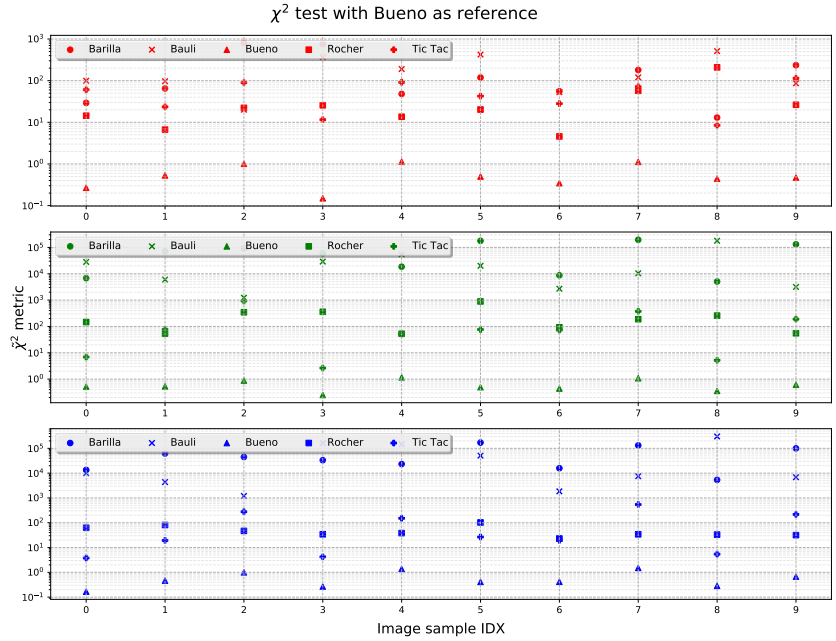



**Figure 4.6:** *Per-channel distances computed with the $\chi^2$ metric for each product 10-image sample using the Bueno average histogram as reference.*

In order to benchmark the customized algorithm developed so far, the same 5-class dataset is employed to train the Custom Vision AI model. In detail, 15 labeled images are given as a training set for each class and a 3-hour time window is guaranteed. The model, however, only takes about 45 minutes for what it considers to be a full training. Afterwards, the same test sets for each class are provided for classification; the performances are detailed in the confusion matrix below, compared for reference with the best-performing metric in the customized dataset.

| Real \ Pred | Barilla | Bauli | Bueno | Rocher | Tic Tac |
|---|---|---|---|---|---|
| **Barilla** | 10 | 0 | 0 | 0 | 0 |
| **Bauli** | 0 | 10 | 0 | 0 | 0 |
| **Bueno** | 0 | 0 | 10 | 0 | 0 |
| **Rocher** | 0 | 0 | 0 | 10 | 0 |
| **Tic Tac** | 0 | 0 | 0 | 0 | 10 |

**Table 4.4:** *Confusion matrix for the best-performing metric in the customized algorithm, the $\chi^2$ test.*

| Real \ Pred | Barilla | Bauli | Bueno | Rocher | Tic Tac |
|---|---|---|---|---|---|
| **Barilla** | 10 | 0 | 0 | 0 | 0 |
| **Bauli** | 0 | 10 | 0 | 0 | 0 |
| **Bueno** | 0 | 0 | 10 | 0 | 0 |
| **Rocher** | 0 | 0 | 0 | 10 | 0 |
| **Tic Tac** | 0 | 0 | 1 | 1 | 8 |

**Table 4.5:** *Confusion matrix for the algorithm trained via Custom Vision AI, the Microsoft ML platform.*

Once again, the sample sizes are of course limited and hence these results do not justify wild generalizations and require some care in interpretation. However, the classification capabilities of Custom Vision AI are indeed excellent and they are comparable to the predictions obtained by the customized algorithm. Moreover, the online service also outputs the prediction percentages and in the only two mis-classified instances those values were about 55%, signaling some uncertainty; in all the other cases, these values were hardly below 90%.

Despite this fact, there are a number of reasons why the customized algorithm is still considered a better choice; in fact, it is:

- ✓ **more controllable**, as Custom Vision AI provides little room for customization and the training mostly happens inside a "black box". The algorithm developed for this analysis, on the contrary, has been thoroughly studied in its different developing stages, which have been planned for this specific use case and are deeply understood; the parameters that control it (for instance, the `threshold` value that modulates the cropping) are well-known and, perhaps even more importantly, can be modified to suit different use cases; lastly, the source code is available and it can be revised at will, at least in principle;

- ✓ **faster**, both in its training and in its test phase. The customized algorithm takes about 10 minutes to complete its training and create the model with the average histograms to perform the classification tasks, while the Microsoft service took about 45 minutes to complete this stage. In addition, when testing the customized algorithm is designed to also take in input folders of images, performing the classification recursively on each file. Custom Vision AI takes instead advantage of an intuitive drag and drop feature, which however only allows the user to input single images[1]; this makes the testing considerably longer and particularly cumbersome;

- ✓ **more versatile**, of course within the ground range of applicability of the algorithm. Custom Vision AI imposes a 4-MB image size cap, which is very limiting even for stan-

[1]By taking advantage of the so-called endpoint – a connection point where HTML files or active server pages are exposed, providing information needed to address a web service – it is actually possible to provide Custom Vision AI with groups of files, bypassing the intrinsic limitations of a drag-and-drop GUI. However, this route requires non-trivial programming capabilities, which make it an obstacle for general use applications.

dard photos taken on the most modern devices. Clearly, no such thing is present in the customized algorithm, which only requires the images to be in a standard .jpg/.jpeg format.

## 4.5 An advanced application: embedding in an industrial framework

Technology readiness levels (TRLs) are a general indicator to estimate the maturity of innovative technologies during the development stages of a program or an industrial project. The primary purpose of using TRLs is to manage the progress of research and implementation activities within an organization and to track them with a formal parameter to measure their advancement status.
At this point of the study, a customized machine learning algorithm has been developed. Its pre-processing stage has been shaped after a detailed study of the images in the dataset, class templates and metrics for classification have been devised and the tests performed have achieved really satisfactory classification results, albeit in a controlled scenario (images displaying a single product on a white background).

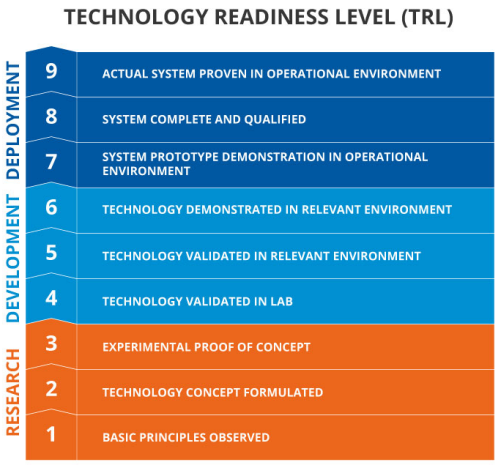


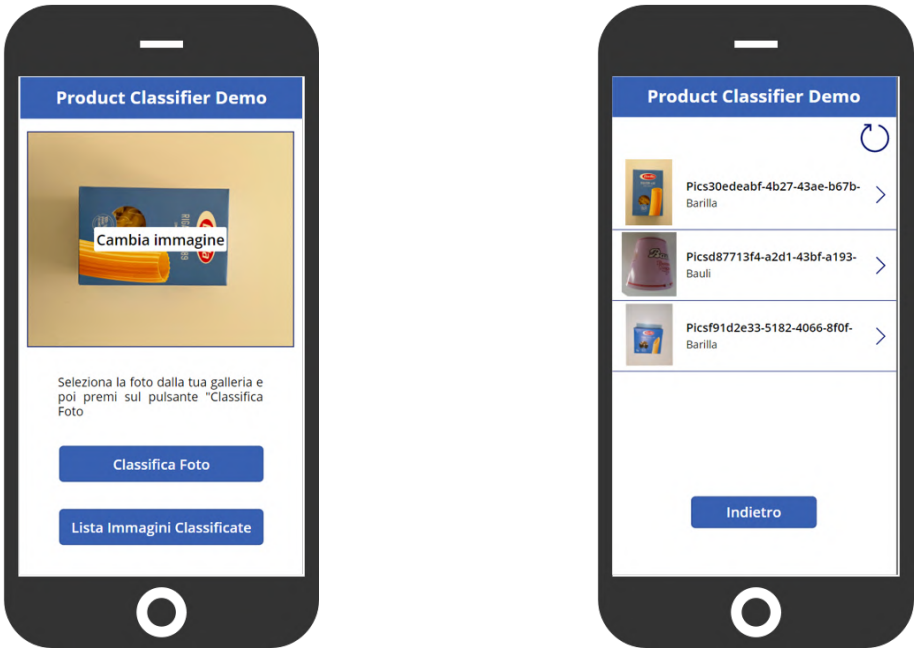

**Figure 4.7:** *Infographic showing the technology readiness levels with the indication of their progress status.*

However, the level of sophistication attained is far from what an industrial solution requires; in order to design a suitable implementation for this kind of reality, the model obtained should be embedded in a more complete architecture that allows a direct and possibly intuitive interaction from the user and a certain degree of automation.

In terms of TRLs, the goal is to accomplish an evolution from level 1 – designated for the stage of basic principles observation – to level 3, where an experimental proof of concept (PoC) is built, meaning a realization of a certain method or idea in order to verify its practical potential and feasibility.

The end result is an application, compatible for desktop and mobile use, which allows to take or load an image and classify it on the basis of the model developed as described previously; the results of the classification are then stored in a list which is also accessible through the app. In Fig. 4.8 the home page and the history page are shown: the design is extremely intuitive and it allows for a graphical interface that forestalls the necessity to resort to the command line. Indeed the visual aspect appears slightly elementary, but it should be made clear that this implementation is meant for an industrial, demonstrative use case and it therefore does not necessarily need the kinds of embellishment of a market-oriented product.

The GUI with which the end user interacts when using the app is actually the most superficial layer of this whole implementation. Beneath it there lies a complex hidden architecture, portrayed in Fig. 4.9: the data collection and visualization stage (a) is the starting/ending point of a much more articulate procedure.
A first necessary item for this machinery to work is a storage for the images, represented in (b). This is the Azure Blob storage (Blob stands for Binary Large Object, a kind of format that requires a specific handling), Microsoft's data storage platform. These images are then kept




(A) *Interface of the home page of the app.* (B) *Interface of the classification history in the app.*

**Figure 4.8:** *Two screenshots showing the interface of the app developed to integrate the classification model in a more structured implementation employing tools from the Microsoft world.*

in this location for a virtually unlimited amount of time and they could potentially serve for statistics or, in future applications, market researches.

In the same location, the JSON file representing the model is stored, together with an Azure function (c). This last entity, in particular, is a serverless compute service that enables a user to run event-triggered code: in simple terms, it runs a script in response to a variety of events. This specific function invokes the model and it performs the classification task on the image.

Clearly, the triggering event for this action to take place needs to be defined: this is the upload of a new image. This aspect is coordinated by Microsoft Power Automate (d), a cloud-based service that gives the chance to create and automate processes and actions with repetitive tasks that save a lot of time and effort. Microsoft Power Automate automates workflows between the services and the app built on top of them, triggering the activation of the Azure function in order for the new image to undergo classification.

This same tool is also the one that, once the classification has been performed, orchestrates the presentation of the results in a SharePoint List (e), a flexible collection of data that can be shared across multiple members and that contains the predicted class, the image and its filename. This list can also be visualized in Microsoft Teams (a collaborative platform for hybrid working serving as a hub for a variety of cloud-based applications), further enhancing the interconnection range of this solution.
The loop is closed back to the app itself, which as noted also shows the classification history as collected in the SharePoint list.

As articulate as it may appear, this system is capable of running smoothly and with perfect harmony between its components; this is clearly a great advantage of the fully-integrated platform offered by Microsoft.

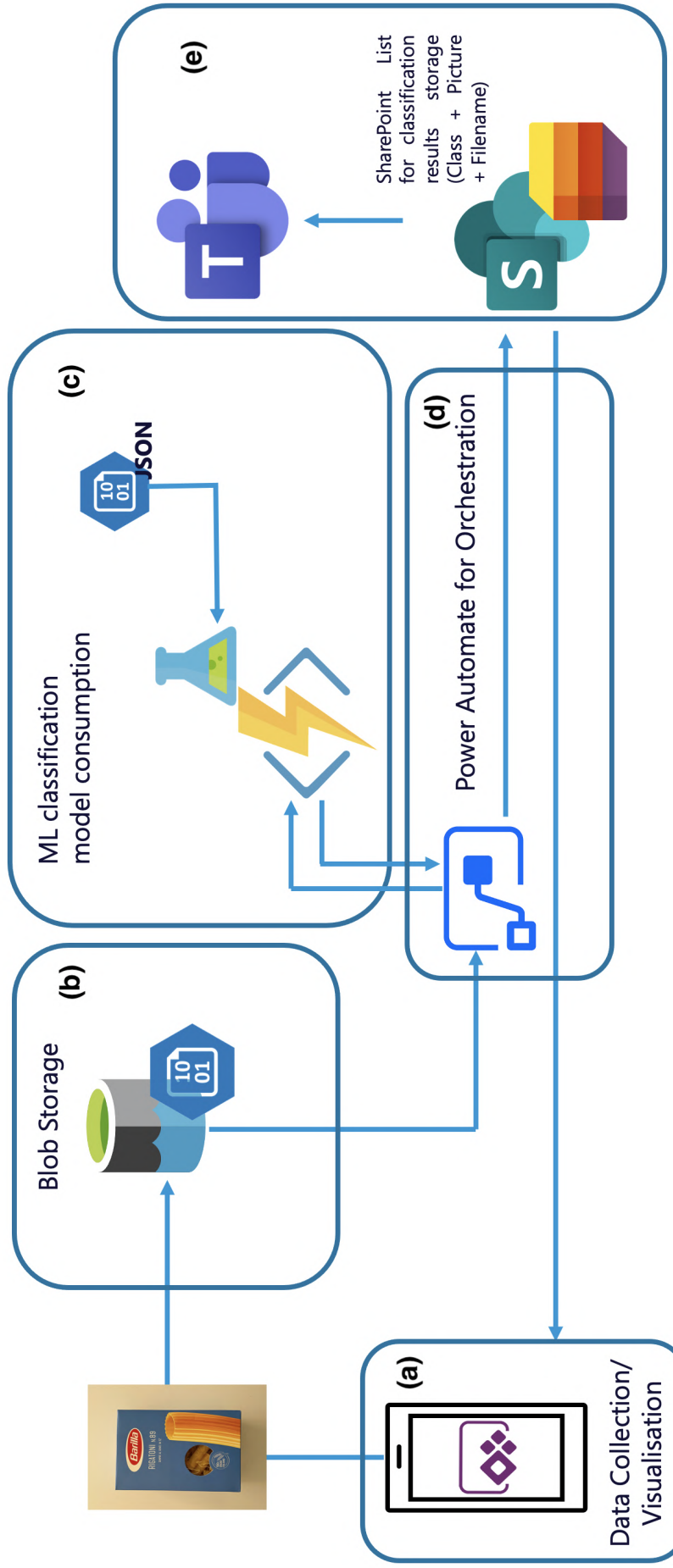



**Figure 4.9:** *Schematic representation of the full architecture behind the development of an application in an industrial environment. The connections between the different sub-structures are shown, as well as the main functions that each fulfills.*

# Chapter 5

# AI for product recognition

WITHIN the framework defined in the toy model, containing mono-product images on a white background, a classification model was eventually developed using a clustering-like logic. This model has been deeply studied, analyzing its pre-processing steps with a quantitative approach that made it possibly to clearly define parameters and to have a well-rounded idea of the operations that the data is undergoing. The end result is a fully-customized model which has proven to be robust and which has shown excellent classification capabilities, although with samples somewhat limited in size; moreover, the model is very compact as far as its storage size is concerned, which is a great advantage especially considering that its training set is composed of images.
The counterpart of these aspects is that this classification model only works effectively in the context in which it has been developed and conditioned, consisting of photos of a single product on an almost white background and primarily with a box-like shape. The model is still far from real-world classification problems and its scope of application has shifted with respect to the initial goals, nonetheless profitable and impacting applications can indeed be found.

At this point, a step forward is required for the classification process to be generalized. The first assumption to be dropped is the requirement that only a single product is shown in an image, as this deeply distances the classification model from reality. Images taken in real supermarkets show a rather broad spectrum of variations, as object appearance tends to display relevant changes in scale, pose, viewpoint, lighting conditions and due to the possible presence of occlusions (obstacles that partially or totally cover the target object). Additional peculiarities characterize the grocery product domain: on the one hand, different products can be very similar and only minor packaging details allow to discriminate them, on the other hand, it is also possible that two packagings of a same product differ considerably over time, for instance because of special offers advertisements or for limited editions.
The second generalization step will be to remove the constraint of a white background, which again is an artifact compared to real-life scenarios.

As it is, this classification task has a high degree of complexity. In order to achieve reasonably satisfying results in such a broadly-defined case study, it becomes binding to fully exploit the capabilities of more advanced algorithms, based on deep learning. By definition, deep learning identifies a class of machine learning algorithms that employ a large number of hidden neuron layers (typically more than 6 but very often much higher than this value, in the tens or hundredths) with nonlinear activation functions to progressively extract higher-level features from the raw input data. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or entire objects.
Artificial neural networks have hence unique capabilities that enable deep learning models to solve tasks that classical machine learning models could never solve.

In particular, to tackle this multi-object detection task, the idea is to employ algorithms

(A) *Pasta Barilla*

(B) *Tonno Riomare*

(C) *Ferrero Rocher*

**Figure 5.1:** *Sample images for each of the considered classes in the analysis of advanced object detection algorithms.*

that take advantage of the so-called single-shot approach. This basically consists in the model only observing the image at once, in a way as a whole, before outputting the detection outcomes guessed in the form of boxes sorrounding the objects the model is trained on. The details will be discussed later on in the chapter, but this technique should in principle entail better time performances at no worrisome accuracy costs.

A focus is then made on the YOLO (you only look once) model, a state-of-the-art implementation which ensures great accuracy detection and especially fast performances, both at training time and in the actual image detection. This model is especially valuable for concrete uses: the efficiency it grants is useful considering the constraints related to costs for cloud usage, whose computational power is necessary to perform the training of such complex algorithms.
In this part of the analysis, three target classes are chosen, representing common goods found in grocery stores and with a distinctive package: Pasta Barilla (the light-blue *Italian wheat* packaging), Tonno Riomare (the classic version) and Ferrero Rocher (to still have a link with the previous chunk of the study. Prototypical images are shown for each product in Fig. 5.1, but of course multiple variations of angle, scale and environmental light are found in the full dataset.

The results of the analysis will be presented, comparing different training configurations for the YOLO model; a best model selection will then be laid out.

---

This chapter expands the analysis performed so far employing more advanced yet less customizable algorithms, with the purpose of generalizing the classification capabilities of the model to images actually taken in a supermarket-like context, where the conspicuous and variegated challenges. Single shot detectors will hence be briefly presented, with a particular focus on the YOLO model, whose implementation details will also be shortly discussed. Then, the analyses carried out with these tools will be outlined together with the preliminary results obtained. Insights and future ways to extend this study will then be discussed.

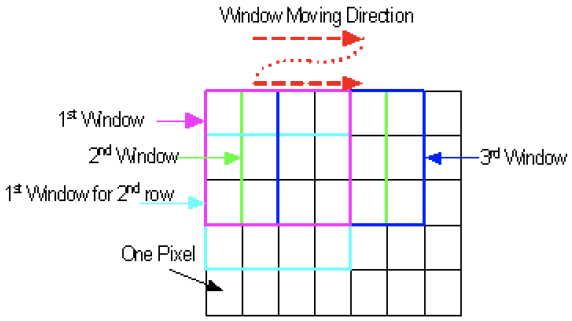
## 5.1 The single-shot approach

Image classification in computer vision represents a task in which an image is taken in input by a certain model (a deep learning one in this instance) which has to predict the membership of objects on which the model has been previously trained. Object detection embodies an evolution of this task in that the algorithm is not only required to predict the class of the objects but also to find them in the images in terms of bounding boxes around their locations. It is hence natural to think of building an object detection model on the top of an image classification model. Once a good image classifier has been developed, the model will only need to be trained in locating what an object is and where it can be found in an image.

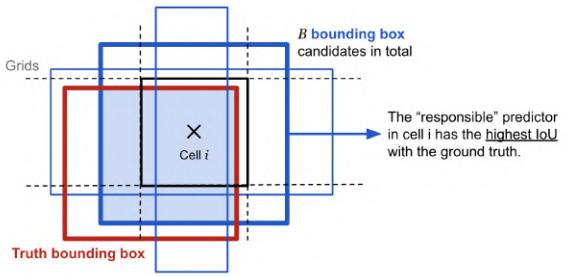
Such a task appears almost instantaneous in human beings, who are gifted with an accurate and fast visual system that makes them capable of performing complex tasks with little conscious thought. This innate ability is very hard to transfer to machines, which require non-trivial implementations.
Possibly one of the most immediate intuitions on how to detect objects is to slide a window across the image and classify whether the image in that window (meaning its cropped out portion) is of the desired type or containes the target object. A visual representation of this kind of detection is shown in Fig. 5.2a, but of course issues such as the size of such window and how to slide it across the image are problems that require ad hoc studies and often empirical solutions.

Even if the basic concept of a sliding window may appear straightforward and fruitful, there are at least two issues that require careful handling in the model implementation. First, there is no way to know the size of the window a priori so that it in fact contains the object(s) inside of it, objects which are likely found at different angles and scale especially in a supermarket scenario. Secondly, the aspect ratio – the ratio between height and width – of the bounding box for a given object is unknown; this could also play a factor in a real-world application, even though it should be noted that goods and food packages tend to have rather fixed proportions, so the scale is more likely to be a predominant factor.

To overcome these challenges, a naïveapproach would be to test different sizes and shapes of sliding window, but this procedure would definitely be very computationally intensive, especially considering that deep neural networks are involved in these tasks.
However, there exists a class of more advanced algorithms – like YOLO (You Only Look Once) and SSD (this is the specific Single-Shot Detector designed by Google) – have developed a rather sophisticated method, known as known as *single-shot approach*[1]. These algorithms use a fully-convolutional approach in which the network is able to find all objects within an image in one pass (hence "single-shot" or "look once") through the convolutional network; the region proposal stage is therefore skipped to achieve higher efficiency.




(A) *Schematic representation of the sliding-window approach for object recognition in images (image freely adapted from [64]).*




(B) *Schematic representation of the single-shot approach for object recognition in images (image freely adapted from [65]).*

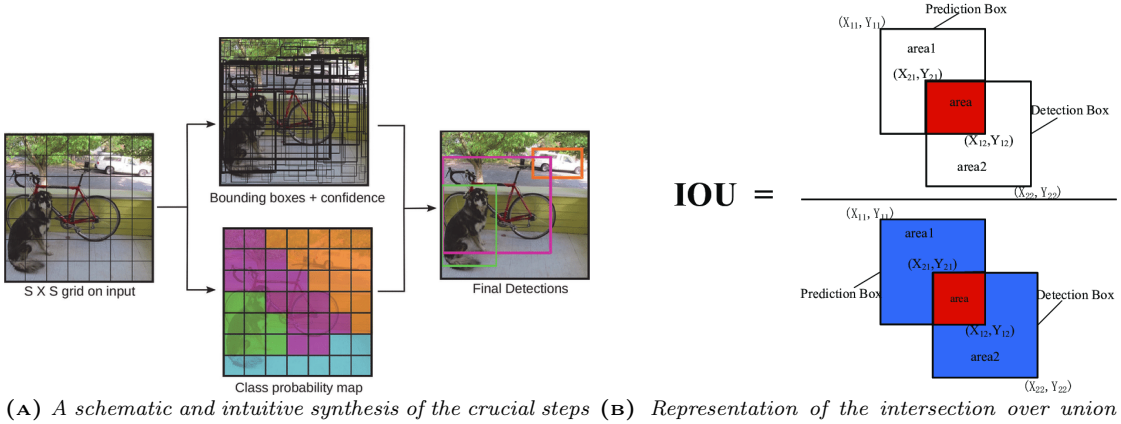
**Figure 5.2:** *Comparison between the typical sliding-window approach employed by convolutional networks to detect objects in an image and the single-shot approach used instead by the implementations considered in this analysis.*

Instead of employing a sliding window, detectors based on single-shot divide the image using a grid and have each grid cell be responsible for detecting objects in that region of the image. The detection itself is carried out by predicting the class and location of an object within that

[1]It should be noted right away that this expression stands for a class of deep learning algorithms and it should not be confused with the acronym SSD (single-shot detector) which instead refers to a specific realization of such algorithm, in an implementation developed by Google in 2016 [63]. Both expressions will appear later on, so caution is made to always make it clear which is the one being referred to.

region. If no object is present, then the image will be considered as background class and the location procedure is disregarded. This is how single-shot detection skips the region proposal stage and yields final localization and content prediction at once [66, 67].

Operationally, the algorithms discretize the output space of bounding boxes into a set of default boxes over different aspect ratios and scales for each image, as shown in Fig. 5.2b. At prediction time, the network generates scores for the presence of each object category in each default box and produces adjustments to the box to better match the object shape. Additionally, the network combines predictions from multiple initial guesses with different resolutions to naturally handle objects of various sizes [68].




**(A)** *A schematic and intuitive synthesis of the crucial steps* **(B)** *Representation of the intersection over union (IoU) of single shot approach chosen for this part of the analysis defined in Eq. 5.1 (image freely adapted from [70]). (image freely adapted from [69]).*

**Figure 5.3**

During training time the default boxes employed for the objects are matched over aspect ratio, location and scale to the ground truth boxes. A selection is then performed on the boxes with the highest overlap with the ground truth bounding boxes, modeled by the quantity known as *intersection over union* (IoU), defined as

$$\text{IoU} = \frac{\text{Area of intersection}}{\text{Area of union}} \tag{5.1}$$

This parameter, as exemplified in Fig. 5.3b, is an intuitive way of measuring the amount of overlap between two bounding boxes, one guessed by the model and one corresponding to the ground truth box; for a perfect prediction IoU = 1, while for a complete miss IoU = 0.

Together with the loss – which is the penalty for a bad prediction, indicating how bad the model prediction power is on the training set (training loss) on the or test set (test loss) – is capable of quantifying the performance of the trained model on the detection task of the objects on which the training has been based.
This allows a rather effective cross-checking methodology when carrying out the best model selection: it can in fact work as a counter-proof that the chosen training configuration is not decreasing its loss due to overfitting on the training data.

While the single-shot approach is the only one considered in this study, possible alternatives to it do exist, for example in the basic evolution to two-shot detection. The two-shot detection model has two stages: region proposal and classification of the regions, which leads to a refinement of the location prediction. However, while two-shot detection models achieve better performance on average, single-shot detection is in the sweet spot of performance and speed and/or resources available.
In addition, single-shot detectors train faster, which, in the concrete context in which this analysis was carried out, allows to efficiently prototype and experiment multiple models without

consuming considerable expenses (which in an industrial setting also translates into cost factors), for example for cloud computing resources. Also, single-shot is a well-suited approach for uses on embedded devices: practical applications need to foresee reasonably fast, real-time object detection on resource-constrained devices, which can all be delivered.

### 5.1.1 The YOLO model

Among the several algorithms which take advantage of the one-shot approach, this analysis will be carried our employing the YOLO model. The YOLO model (acronym for "You Only Look Once", developed in 2016 [71]) is the very first attempt at building a fast, virtually real-time object detector which also preserves strong detection capabilities.

YOLO has 24 convolutional layers followed by 2 fully connected layers (Fig. 5.4). Some convolution layers use $1 \times 1$ reduction layers in alternation so as to reduce the depth of the features maps: because even low dimensional embeddings contain a lot of information about the starting image patch, this decrease in dimensionality still retains its salient features, without critical loss of information. Not only does this downsampling allow the network to train how to reduce the dimension in the most efficient way, but it is also pivotal in the time performance optimization that makes this model so fast at detection stage.

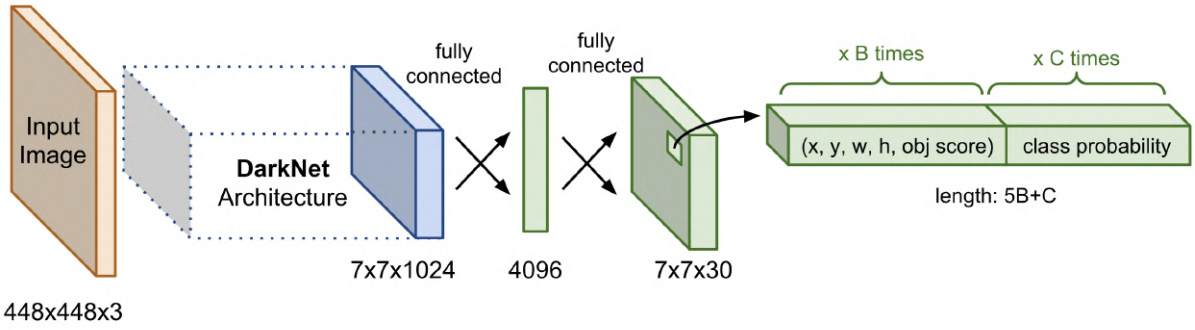



**Figure 5.4:** *The network architecture of YOLO in its original implementation (image freely adapted from [65]).*

Because here $S = 7$, in the last convolution layer a tensor with shape $(7, 7, 1024)$ is obtained in output; this tensor is then flattened. If for each grid cell B bounding boxes are predicted and there are a total of C class probabilities, then, using 2 fully connected layers as a form of linear regression, $7 \times 7 \times (5B + C)$ parameters are retrieved and combined in a $(7, 7, (5B + C))$-shape tensor.

Without wandering excessively in dispersive technical details, the intuition behind YOLO is to re-frame the object detection task as a regression problem between spatially-separated bounding boxes and their associated probabilities of class membership. A single neural network predicts both quantities directly in one evaluation (hence the one-shot approach), skipping the region proposal step and only predicting over a limited number of bounding boxes; this greatly reduces the inference times.

Operationally, still referencing to Fig. 5.4 to have a visual representation of the architecture, YOLO divides the input image into an $S \times S$ grid, with each grid cell assigned to the prediction of only one object thorough a fixed number of boundary boxes.
YOLO predicts multiple bounding boxes for each grid cell, but in order to compute the loss for the true positive, only one of them needs to be selected as representative for the detected object, if that is the case for the specific image fed to the network; this bounding box is the one with the highest IoU with the ground truth. Such a strategy leads to a specialization in bounding box prediction which in turn strengthens the model detection power with respect to certain sizes and aspect ratios [72].

To obtain a quantitative value for the loss, YOLO uses sum-squared error between the predictions and the ground truth. The analytical formula of this quantity is cumbersome and it deviates from the scope of this presentation, so it is left out. Just to give an intuition, this loss can be written as the sum of two terms, a *localization loss* devoted to quantifying the errors between the predicted boundary box and the ground truth (establishing how well the object has been located), and a *classification loss* which instead acts as a penalization for a mismatch between predicted and real class memberships (evaluating the classification error).
Two real-valued scale parameters defined in $[0, 1)$ tune the respective weight of these penalizations, in order to establish how each penalty influences the overall process. In this analysis, both variables are left untouched at their native value of 0.5, meaning the two contributions to the loss are weighted equally.

Defined like so, this loss function treats errors the same regardless of the location in which they occur. A small error in a large bounding box is generally negligible, but it could have a much greater impact (on IoU, for instance, but also on the general detection capabilities) if found in a small box.
As a matter of fact, YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts a fixed number of them: this constraint limits the number of nearby objects that the model can predict. Because of that, YOLO struggles with small objects, especially those that appear in groups (such as flocks of birds). Also, it tends to be less efficient when it is required to generalize to objects in new or unusual aspect ratios or configurations or with an irregular shape, since it learns to predict bounding boxes from data.

However, several releases have been developed to tackle these shortcomings and further improve the capabilities of YOLO. In its second version, YOLOv2, $k$-mean clustering is employed on the training data to find good priors on the size of the bounding box, taking advantage of the spatial correlations that almost always exist between the objects depicted in an image [73].
In YOLOv3, a further development, a confidence score is predicted for each bounding box using logistic regression, while YOLO and YOLOv2 employed the sum of squared errors. YOLOv3 also adds cross-layer connections between two prediction layers (except for the output layer) and earlier finer-grained feature maps, making it more robust in the detection of small objects and showing fewer false positives in background areas [74].

Because of its extremely reduced detection time and the accuracy it nonetheless grants in testing, the YOLO model and its future improvements are also the algorithms of choice when dealing with even more complex scenarios.
Just to give an idea of the most exotic applications, in [75] for instance a version of YOLO specifically built to optimize speed was employed to perform real-time embedded object detection in videos; an average prediction confidence of around 60% was reached (which is still more than twice the accuracy of typical real-time algorithms), with a run-time 3.3× faster compared to basic YOLO implementations. Moreover, in [76], YOLO was also employed to achieve 3D object recognition, meaning the algorithm identifies the object in all its three spatial dimensions and not just through a flat bidimensional box; because a higher dimensionality implies a significant additional computational burden, the time optimization of YOLO (a purposefully modified version) once again proves to be decisive.

## 5.2 Analysis and results

This analysis will specifically employ the third release of the YOLO model, known as YOLOv3 and developed in 2018 by the same team of developers [74].
In order to work with this pre-built and partially pre-trained model, the `ImageAI` library is taken advantage of [77]. It offers self-contained deep learning tools that allow to employ existing models and pre-trained configurations for personalized applications and with custom-made datasets.

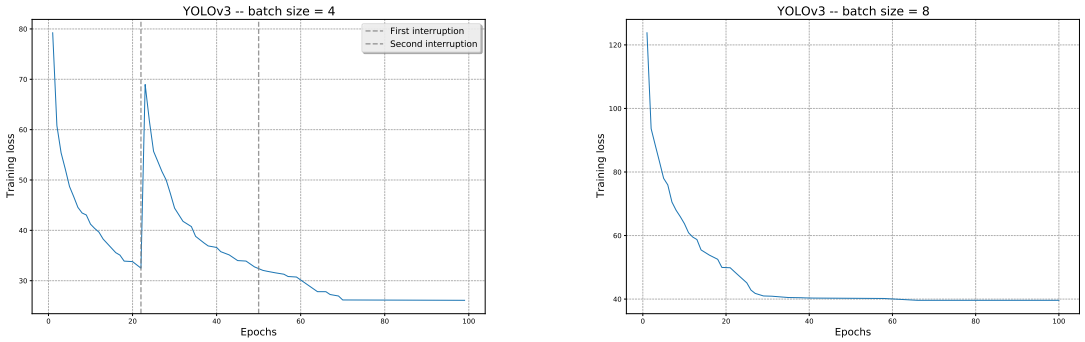
Regardless of how complex and advanced the model may be, the training procedure must begin by labeling images with the target objects for detection. This operation, which can not prescind from human intervention, is needed so that the model is able to train on unseen objects. It basically only consists of drawing boxes around the targets in order to create a labeled training and validation set; the `LabelImg` interfaces allows to do so in a rather intuitive way, automatically creating constraint files that the YOLO then employs to translate the graphical information of the drawn boxes into a readable format for its architecture (known as Pascal VOC format).

Because the model architecture is already defined, there are no degrees of freedom that can be explored in this respect; the present analysis rather focuses on the definition of an efficient and effective training procedure.
There are two parameters that can be tuned:

- the *number of epochs*, also referred to as number of experiments, corresponds to the number of times the network trains on the training set;
- the *batch size*, number of samples that will be propagated through the network at each iteration. The use of batches decreases memory usage, because the network is trained using fewer samples at one time; however, the smaller the batch the less accurate the estimate of the gradient tends to be.

Considering that `ImageAI` gives the chance to resume training at any given point, for these exploratory studies it is set $n_{epochs} = 100$. As far as the batch size is concerned, the documentation suggests to employ values no less than 4 and possibly that are multiples of 8. However, configurations with a high batch size (namely more than 16) imply a computational burden that the cloud is not capable of coping with, at least with the infrastructures at disposal for this analysis. It is hence decided to compare and contrast configurations with $n_1 = 4$ and $n_2 = 8$ as batch size.

The trend of the training loss for both configurations over the 100 epochs is displayed in the plots below.




**(A)** *Training loss of the YOLO model with batch size = 4.* **(B)** *Training loss of the YOLO model with batch size = 8.*

**Figure 5.5:** *Training loss trends for the two model training configurations considered in this analysis.*

In (A), the two dashed lines show the two interruptions that the training has undergone; the spike registered around epoch 22 is probably due to some unforeseeable effect caused by the training interruption, which instead leaves the loss unaltered in the second occurrence. However, this does not appear to be an obstacle to the training procedure, which clearly reaches convergence way before its final epoch (the loss becomes pretty much flat around epoch 70).
Convergence in the training loss is also achieved by the model with batch size = 8 (B), slightly before than the previous (a stable value is reached already at epoch 40, with a further, slight

decrease after epoch 60). In this case, the trend is smoother but it can be noticed how in (B) the final loss reached is higher by around 45% with respect to (A).

When the training is perfected, test images – unlabeled images that were never seen by the model itself – are selected to verify the detection and classification performances of the models obtained. While it can be very hard and intricate to quantify how well the model is detecting the objects[2], it is still possible to appreciate the detection capabilities of the trained YOLO model.


(A) *Batch size = 4*

(B) *Batch size = 8*

(C) *Batch size = 4*

(D) *Batch size = 8*

**Figure 5.6:** *Two examples of the impressive detection performances of both configurations for the Barilla class.*

For instance, in Fig. 5.6, two photos are shown with the respective Barilla packages detected in both training configurations. The results are impressive: in (A) and (B), in particular, the model is capable of retrieving basically all the target packages (it should be kept in mind that the model was only trained on the light-blue packaging), even those that are arranged sideways. As a general trend, the configuration with the higher batch size tends to show more misclassifications, while the only two errors in (A) come from partially cropped packages, which in a way justifies the possibility of a mistaken classification.
Very satisfying results are also shown in (C) and (D), an image which poses the additional

[2]A formal analysis in this regard would require at a minimum a manual count of all the objects in each test image; however, some sort of benchmark would also be needed, so for instance a human control group should be employed to retrieve average human detection values for the test set sample and compare them with the selected models. At this exploratory stage, however, a qualitative comparison will still be able to convey a general idea of the model under scrutiny.

challenge of displaying the aisle at an angle. Nonetheless, both configurations are still capable of detecting the large majority of the Barilla products, again with the model trained with a batch size = 4 having a slight overhead.

Some problems arise when dealing with the other two categories, which are often either confused between each other, or not distinguished correctly as individual items or completely missed in the detection altogether.

Referring to 5.7, in (A) only very few chocolates boxes are recognized, but in these cases the shape is usually closely followed. There is however a consistent ambiguity in assigning the class membership, so much in fact that at times a double bounding box surrounds the same box; a partially comforting factor lies in the fact that accuracies for the Riomare class are almost in all cases lower than those of the Rocher. In (B), where batch size = 8, the detection is actually not so poor, in that the box shape is faithfully recognized and classified; the presence of a fully non-contextualized Tonno Riomare is however proof highlighting some weakness in the model. In (C), a single package is detect but the ones in foreground are completely skipped; when they are in fact considered (D), though, the model appears to struggle in identifying the correct shape of a single item and some of these tuna boxes are actually detected as Ferrero Rocher.

In these instances, there appears not to be a better configuration between the two. Overall, the model with smaller batch size seems more conservative in detection, managing to omit an unreliable guess more often compared to the other; aside from that, also this configuration is not exempt from misclassifications.

It can be hard to fully understand the difference in detection and prediction performances between the different product classes. The most reasonable hypothesis is that the Barilla package has a definite and sharp package, together with a fairly distinctive pattern on its front. This in particular is a feature that the model surely picks up on at training time and that makes this product highly distinguishable from background. This is not entirely true for the other two packagings, which in addition are frequently organized in their aisles so that it is hard to discern each box.

## 5.3 Future insights

It has been seen how the analysis carried out in this section has led to promising results, but at the same time has shown some quite disruptive misclassification and misdetection that indeed require some careful consideration.

The most intuitive, "bottom-up" approach is to provide the model with a larger training set, especially for Riomare and Rocher, the two classes which have shown the higher number of errors and lower overall prediction confidence. With more images, it will be easier for the model to better understand for instance where the borders of the tuna packaging are, in order to discriminate different items and associate a bounding box to each; also, it is expected that the confusion between Rocher and Riomare should gradually be reduced.
Still, obtaining more training images requires also their manual labeling, which as discussed is a rather time-consuming and mechanical task. A more radical solution might be ideal to qualitatively improve the classification stage of the YOLO model.

A possibility in this regard could be to exploit the concept of *proximity*, which by the way is somewhat at the core of the customized part of the analysis as well. Goods in a supermarket are clearly not arranged randomly: different aisles cluster the different categories and, within sight, products of the same genre are usually found. That is to say, it is very unlikely that a Ferrero Rocher box is found among tuna tins or viceversa. Accounting for this factor would firstly remove these spurious identifications, even if it is not a process that would improve the detection directly. The desirable byproduct, though, could be that the model would then rightly classify also other surrounding objects from the initial, more accurately detected guesses.

Of course, in a more realistic use-case, where the products to be detected and classified

(A) *Batch size = 4* (B) *Batch size = 8*

(C) *Batch size = 4* (D) *Batch size = 8*

**Figure 5.7:** *Two examples of the issues encountered in detection and classification for the Tonno Riomare and the Ferrero Rocher classes, for both batch sizes.*

are multiple (likely in the order of tens), the selection of the "ground truth" that defines what location of the store the one depicted in a given image actually is would become a crucial step. The model, in practice, needs to be reasonably confident about its highest-confidence guesses, as with this implementation they could impact the ones that follow. In an end-user oriented application it would not appear unreasonable to give the user the capability of manually inputing a couple of initial detections, even if this would hinder the full automation of the algorithm.

# Conclusions and outlooks

FINAL goal of this analysis was to develop an algorithm to perform detection and classification of products in the retail context.

In the first part of the study, a toy model was defined so as to have a clearer insight of the data under investigation; in it, 5 product classes are considered under the only requirement that a single product on a white background has to be present. Through a color-based analysis, a novel approach to perform feature extraction from RGB histograms was developed, together with a customized method to create reference classes and different metrics that were compared to measure the distance in a sort of $k$-NN-like algorithm. In this sense, the $\chi^2$ metric was found to have the best performances.

Also the practical implementation of the model adds a quite powerful advantage to it. The classification procedure follows a classification logic, but the employment of average histograms makes it independent from the training set size: unlike typical clustering algorithm, which iterate on all training data points, this model only needs to compare a sample to a number of average histograms corresponding to the number of product classes. This significantly reduces computation times both in training and testing and it is also extremely convenient memory-wise, as only few kilobytes are needed after training compared to the several megabytes required for each image, which can be a burdensome data type to store when in large numbers.
These strengths were especially highlighted in the comparison with Custom Vision AI, an image recognition service provided by Microsoft which represents a ready-to-use benchmark in this field. The two classification models have similarly very good performances, but the customized one relies on a well-known and controllable procedure, which in addition offers a higher degree of flexibility in terms of both image size and in classification times.

The restrictions imposed in the definition of the toy model do not impose critical limitations on the applicability of the algorithm, which in an industrial context is always a prime factor to be considered; these applications can be even conceptually far from the idea on which the algorithm was initially based. For instance, any implementation that requires a visual search approach can benefit from it: a similarity search can be performed in order to determine the class of that given object with a duly trained model and through a reliable procedure. Several other possibilities do exist, ranging from quality control of any color-based industrial productive line (test of color requirements, components recognition or sorting, ...) to shelf management in retail.

In the second part of the analysis, all the initial assumptions were dropped in search for a general-purpose algorithm to be used for item detection and classification, this time considering actual images from supermarkets or grocery stores but again limiting the task to three product classes.
Because of all the discussed challenges this scenario imposes, more advanced models were examined, focusing in particular on a single-shot approach where the model considers each image as a whole and detects the objects in it with a unique attempt (i.e: without employing a sliding window or performing a region proposal step). The algorithm of choice was YOLO, a deep convolutional neural network which proved to obtain satisfactory accuracy values at a very

limited computational cost.

The results obtained are very promising, but the model still shows some weaknesses in detection or classification of some products. In particular, common shortcomings encountered are related to a confusion between the two items or to a flaw in correctly detecting the shape of each individual product.
In general, this exploratory analysis has indeed laid the foundations for future improvements: they can vary from a straightforward enlargement of the training set, which however requires the bottle-neck operation of manual labeling of the images, to the implementation of more sophisticated auxiliary techniques to refine detection and classification. A possibility, for instance, could be to exploit a further similarity analysis on top of the YOLO model. The precise criteria with which goods are usually organized in a supermarket could be taken advantage of to filter the guesses output by the neural network.

In addition to that, in the future it could be possible to extend the analysis performed with the single-shot approach and, once robust results are obtained in the detection task, build the customized ML classification algorithm developed in the first part on top of it. This will in fact provide classification guesses for each detected box requesting minimal storage support (only the average RGB histograms for each product are required) and granting well-known and validated procedures. The overall result could be a possibly rather new example of an ensemble model approach for object detection and classification.

# Bibliography


[1] S. Whiteson and D. Whiteson, "Machine learning for event selection in high energy physics," *Eng. Appl. of AI*, vol. 22, pp. 1203–1217, 12 2009.

[2] J. Brehmer, K. Cranmer, I. Espejo, A. Held, F. Kling, G. Louppe, and J. Pavez, "Constraining effective field theories with machine learning," *EPJ Web of Conferences*, vol. 245, p. 06026, 01 2020.

[3] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel, A. Aurisano, K. Terao, and T. Wongjirad, "Machine learning at the energy and intensity frontiers of particle physics," *Nature*, vol. 560, 08 2018.

[4] P. Andrade, T. Bell, J. Eldik, G. Mccance, B. Panzer-Steindel, M. Santos, S. and, and U. Schwickerath, "Review of cern data centre infrastructure," *Journal of Physics Conference Series*, vol. 396, pp. 2002–, 12 2012.

[5] CERN-DataCenter. (2018, 08) Key facts and figures. Accessed: 2021-01-20. [Online]. Available: https://information-technology.web.cern.ch/sites/information-technology.web.cern.ch/files/CERNDataCentre_KeyInformation_01June2018V1.pdf

[6] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature communications*, vol. 5, p. 4308, 07 2014.

[7] P. Hamet and J. Tremblay, "Artificial intelligence in medicine," *Metabolism*, vol. 69, 01 2017.

[8] L. Gatzoulis and I. Iakovidis, "Wearable and portable ehealth systems," *Engineering in Medicine and Biology Magazine, IEEE*, vol. 26, pp. 51 – 56, 10 2007.

[9] K. Theofilatos, N. Pavlopoulou, C. Papasavvas, S. Likothanassis, C. Dimitrakopoulos, E. Georgopoulos, C. Moschopoulos, and S. Mavroudi, "Predicting protein complexes from weighted protein–protein interaction graphs with a novel unsupervised methodology: Evolutionary enhanced markov clustering," *Artificial Intelligence in Medicine*, vol. 63, 02 2015.

[10] A. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, pp. 1–5, 01 2020.

[11] J. He, S. Baxter, J. Xu, J. Xu, X. Zhou, and K. Zhang, "The practical implementation of artificial intelligence technologies in medicine," *Nature Medicine*, vol. 25, 01 2019.

[12] A. Ramesh, C. Kambhampati, J. Monson, and P. Drew, "Artificial intelligence in medicine," *Annals of the Royal College of Surgeons of England*, vol. 86, pp. 334–8, 10 2004.

[13] E. Kleinpeter, "Four ethical issues of "e-health"," *IRBM*, vol. 38, 09 2017.

[14] L. Anaya, A. Alsadoon, N. Costadopoulos, and P. Prasad, "Ethical implications of user perceptions of wearable devices," *Science and Engineering Ethics*, vol. 24, pp. 1–28, 02 2018.

[15] S. O. Hansson, "Implant ethics," *Journal of medical ethics*, vol. 31, pp. 519–25, 10 2005.

[16] J. Moor and J. Weckert, "Nanoethics: assessing the nanoscale from an ethical point of view," vol. ISBN, pp. 1–58 603, 01 2004.

[17] J. Nam, S. Park, E. J. Hwang, J. Lee, K.-N. Jin, K. Lim, T. Vu, J. Sohn, S. Hwang, J. M. Goo, and C. M. Park, "Development and validation of deep learning–based automatic detection algorithm for malignant pulmonary nodules on chest radiographs," *Radiology*, vol. 290, p. 180237, 09 2018.

[18] J. Mushtaq, R. Pennella, S. Lavalle, A. Colarieti, S. Steidler, C. Martinenghi, D. Palumbo, A. Esposito, P. Rovere-Querini, M. Tresoldi, G. Landoni, F. Ciceri, A. Zangrillo, and F. Cobelli, "Initial chest radiographs and artificial intelligence (ai) predict clinical outcomes in covid-19 patients: analysis of 697 italian patients," *European radiology*, vol. 31, 09 2020.

[19] IRCCS-Ospedale-San-Raffaele. (2020) Ai-score, a project to calculate prognostic risk from covid-19. Accessed: 2021-03-28. [Online]. Available: https://research.hsr.it/en/news/ai-score-a-project-to-calculate-prognostic-risk-from-covid-19.html

[20] Porini. (2020) Today ai-score was launched, the project to calculate risk from covid-19. Accessed: 2021-03-28. [Online]. Available: https://www.porini.it/news/today-was-launched-ai-score-the-project-to-calculate-risk-from-covid-19/

[21] E. Bedolla-Montiel, L. Padierna, and R. Castaneda-Priego, "Machine learning for condensed matter physics," *Journal of Physics Condensed Matter*, vol. 33, p. 053001, 11 2020.

[22] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, A. von Lilienfeld, K.-R. Müller, and A. Tkatchenko, "Machine learning predictions of molecular properties: Accurate many-body potentials and non-locality in chemical space," *The Journal of Physical Chemistry Letters*, vol. 6, pp. 2326–2331, 06 2015.

[23] J. Carrasquilla and R. Melko, "Machine learning phases of matter," *Nature Physics*, 05 2016.

[24] J. Carrasquilla, "Machine learning for quantum matter," *Advances in Physics: X*, vol. 5, p. 1797528, 01 2020.

[25] A. Grover, A. Kapoor, and E. Horvitz, "A deep hybrid model for weather forecasting," 08 2015, pp. 379–386.

[26] A. Salman, B. Kanigoro, and Y. Heryadi, "Weather forecasting using deep learning techniques," 10 2015, pp. 281–285.

[27] K. Rasouli, W. Hsieh, and A. Cannon, "Daily streamflow forecasting by machine learning methods with weather and climate inputs," *Journal of Hydrology - J HYDROL*, vol. 414-415, 01 2012.

[28] X. Ding, T. Liu, J. Duan, and J.-Y. Nie, "Mining user consumption intention from social media using domain adaptive convolutional neural network," in *AAAI*, 2015.

[29] R. Kaliyar, A. Goswami, P. Narang, and S. Sinha, "Fndnet- a deep convolutional neural network for fake news detection," *Cognitive Systems Research*, vol. 61, 06 2020.

[30] Z. Allam and A. Z. Dhunny, "On big data, artificial intelligence and smart cities," *Cities*, vol. 89, pp. 80–91, 01 2019.

[31] Z. Ullah, F. Al-Turjman, L. Mostarda, and R. Gagliardi, "Applications of artificial intelligence and machine learning in smart cities," *Computer Communications*, vol. 154, 03 2020.

[32] A. Shahat, "A novel big data analytics framework for smart cities," *Future Generation Computer Systems*, vol. 91, 07 2018.

[33] W. Ebeling and T. Pöschel, "Entropy and long-range correlations in literary english," *EPL (Europhysics Letters)*, vol. 26, 09 1993.

[34] H. Hassan, A. Aue, C. Chen, V. Chowdhary, J. Clark, C. Federmann, X. Huang, M. Junczys-Dowmunt, W. Lewis, M. Li, S. Liu, T.-Y. Liu, R. Luo, A. Menezes, T. Qin, F. Seide, X. Tan, F. Tian, L. Wu, and M. Zhou, "Achieving human parity on automatic chinese to english news translation," 03 2018.

[35] Microsoft-TranslatorBlog. (2019, 06) Neural machine translation enabling human parity innovations in the cloud. Accessed: 2021-01-25. [Online]. Available: https://www.microsoft.com/en-us/translator/blog/2019/06/17/neural-machine-translation-enabling-human-parity-innovations-in-the-cloud/

[36] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," vol. 2, 01 2010, pp. 1045–1048.

[37] R. Staudemeyer and E. Morris, "Understanding lstm – a tutorial into long short-term memory recurrent neural networks," 09 2019.

[38] Colah's-Blog. (2015) Understanding lstm networks. Accessed: 2021-01-25. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[39] World-Economic-Forum. (2019, 06) What is the fourth industrial revolution? Accessed: 2021-01-30. [Online]. Available: https://www.weforum.org/agenda/2016/01/what-is-the-fourth-industrial-revolution/

[40] G. Li, Y. Hou, and A. Wu, "Fourth industrial revolution: technological drivers, impacts and coping methods," *Chinese Geographical Science*, vol. 27, pp. 626–637, 08 2017.

[41] M. Xu, J. David, and S. Kim, "The fourth industrial revolution: Opportunities and challenges," *International Journal of Financial Research*, vol. 9, p. 90, 02 2018.

[42] M. Saturno, V. Pertel, and F. Deschamps, "Proposal of an automation solutions architecture for industry 4.0," 07 2017.

[43] A. Petrillo, F. De Felice, R. Cioffi, and F. Zomparelli, *Fourth Industrial Revolution: Current Practices, Challenges, and Opportunities*, 02 2018.

[44] L. Caruso, "Digital innovation and the fourth industrial revolution: epochal social changes?" *AI & SOCIETY*, vol. 33, 08 2018.

[45] M. Mohamed and P. Weber, "Trends of digitalization and adoption of big data & analytics among uk smes: Analysis and lessons drawn from a case study of 53 smes," 03 2020.

[46] S. Wang and H. Wang, "Big data for small and medium-sized enterprises (sme): a knowledge management model," *Journal of Knowledge Management*, vol. ahead-of-print, 04 2020.

[47] T. Kramp, R. Kranenburg, and S. Lange, *Introduction to the Internet of Things*, 09 2013, pp. 1–10.

[48] Y.-K. Chen, "Challenges and opportunities of internet of things," pp. 383–388, 01 2012.

[49] S. Coleman, R. Goeb, G. Manco, A. Pievatolo, X. Tort-Martorell, and M. Reis, "How can smes benefit from big data? challenges and a path forward: S. coleman et al." *Quality and Reliability Engineering International*, vol. 32, 10 2016.

[50] A. Soroka, Y. Liu, L. Han, and M. S. Haleem, "Big data driven customer insights for smes in redistributed manufacturing," *Procedia CIRP*, vol. 63, pp. 692–697, 12 2017.

[51] Business-Model-Sinc. Exploring big data business models & the winning value propositions behind them. Accessed: 2021-02-06. [Online]. Available: https://www.businessmodelsinc.com/big-data-business-models/

[52] V. Nissen, *Digital Transformation of the Consulting Industry—Introduction and Overview*, 01 2018, pp. 1–58.

[53] H. Chesbrough, "Business model innovation: Opportunities and barriers," *Long Range Planning*, vol. 43, pp. 354–363, 2010.

[54] A. Franco, D. Maltoni, and S. Papi, "Grocery product detection and recognition," *Expert Systems with Applications*, vol. 81, 03 2017.

[55] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, "Deep learning for logo recognition," *Neurocomputing*, 01 2017.

[56] D. McCamy, Marcus, "A color-rendition chart," *Journal of Applied Photographic Engineering*, vol. 2, 1976.

[57] A. Broadbent, "A critical review of the development of the cie1931 rgb color-matching functions," *Color Research & Application*, vol. 29, pp. 267 – 272, 08 2004.

[58] A. Clark, "Pillow (pil fork) documentation," 2015. [Online]. Available: https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf

[59] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[60] S. H. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. T. H. Romeny, and J. B. Zimmerman, "Adaptive histogram equalization and its variations," *Graphical Models graphical Models and Image Processing computer Vision, Graphics, and Image Processing*, vol. 39, pp. 355–368, 1987.

[61] J. B. Zimmerman, S. M. Pizer, E. V. Staab, J. R. Perry, W. McCartney, and B. C. Brenton, "An evaluation of the effectiveness of adaptive histogram equalization for contrast enhancement," *IEEE Transactions on Medical Imaging*, vol. 7, no. 4, pp. 304–312, 1988.

[62] A. Reza, "Realization of the contrast limited adaptive histogram equalization (clahe) for real-time image enhancement," *VLSI Signal Processing*, vol. 38, pp. 35–44, 08 2004.

[63] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg, "Ssd: Single shot multibox detector," vol. 9905, 10 2016, pp. 21–37.

[64] Logan-Z. (2013) Computer vision implementation. Accessed: 2021-03-23. [Online]. Available: https://sites.google.com/site/hsi2013logan99/computer-vision/implementation

[65] L. Weng, "Object detection part 4: Fast detection models," *lilianweng.github.io/lil-log*, 2018. [Online]. Available: http://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html

[66] T. Wang, R. Anwer, H. Cholakkal, Y. Pang, L. Shao, and F. Khan, "Learning rich features at high-speed for single-shot object detection," 01 2020.

[67] T. Kong, F. Sun, H. Liu, Y. Jiang, and J. Shi, "Consistent optimization for single-shot object detection," 01 2019.

[68] S. Zhang, L. Wen, Z. Lei, and S. Li, "Refinedet++: Single-shot refinement neural network for object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. PP, pp. 1–1, 04 2020.

[69] R. Magalhães and H. Peixoto, *Object Recognition Using Convolutional Neural Networks*, 11 2019.

[70] S. Cheng, K. Zhao, and D. Zhang, "Abnormal water quality monitoring based on visual sensing of three-dimensional motion behavior of fish," *Symmetry*, vol. 11, p. 1179, 09 2019.

[71] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 06 2016, pp. 779–788.

[72] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, "Yolo nano: a highly compact you only look once convolutional neural network for object detection," 10 2019.

[73] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 07 2017, pp. 6517–6525.

[74] ——, "Yolov3: An incremental improvement," 04 2018.

[75] M. J. Shafiee, B. Chywl, F. Li, and A. Wong, "Fast yolo: A fast you only look once system for real-time embedded object detection in video," *Journal of Computational Vision and Imaging Systems*, vol. 3, 09 2017.

[76] X. Zhao, H. Jia, and Y. Ni, "A novel three-dimensional object detection with the modified you only look once method," *International Journal of Advanced Robotic Systems*, vol. 15, p. 172988141876550, 03 2018.

[77] DeepQuest-AI. (2020) Official english documentation for imageai. Accessed: 2021-03-25. [Online]. Available: https://imageai.readthedocs.io/en/latest/#

*Così, tra questa*
*immensità s'annega il pensier mio:*
*e il naufragar m'è dolce in questo mare.*