Politecnico di Torino

Politecnico di Torino

Master of science in Engineering and Management
Academic Year 2022/2023
Graduation session December 2023

Digital Twin and Machine Learning in welding: implementation of a CNN model for image classification and quality monitoring

Supervisor:
Giulia Bruno
Co-supervisors:
Emiliano Traini
Gabriel Antal

Candidate:
Giorgia Ficili

Acknowledgements

Completing this thesis work has been challenging yet deeply rewarding, and I couldn't have undertaken this journey without the support and guidance of many people around me.
First and foremost, I would like to express my deepest gratitude and appreciation to my dedicated supervisor professor, Giulia Bruno, for giving me the possibility to develop such interesting and enriching topics, and for her invaluable insights and suggestions which have been essential for this research.
I'm also grateful to my co-supervisors, Emiliano Traini, and Gabriel Antal since this endeavor would not have been possible without their constant assistance and full-time availability to help me whenever I needed it.
Lastly, my sincere thanks are to be extended to my family that has been constantly present over my whole course of study, since the very first day. They believed in me at each step of the way, keeping my spirits and motivation high even during times when I myself doubted my own ability to achieve success.

Thank you all for your essential contribution to this intense and enlightening academic journey.

Giorgia Ficili

Abstract

Digital twin technology is rapidly evolving within the Industry 4.0 realm and is emerging as a powerful tool for companies to transform themselves and proceed towards digitalization of their operations. Often, digital twin technology embeds some artificial intelligent models, in the attempt of achieving a so-called 'Intelligent Digital Twin', with even more enhanced capabilities of data analysis, fault detection, decision-making support, prediction, and many more.

In line with the advantages coming from the synergy between digital twin and advanced deep learning algorithms, a Convolutional Neural Network has been developed within this thesis work, thought as a component of a broader Digital Twin comprehensive framework for Resistance Spot Welding quality monitoring. Indeed, the purpose of the work performed is to give a contribution in trying to address the quality issues impacting Resistance Spot Welding process, which is often affected by some welding defects. Particularly, the focus is on a specific and quite common defect in RSW, i.e., expulsion: it consists in the ejection of molten metal out of the nugget area during the welding process and can strongly affect the quality of the welding in terms of joint strength and other structural issues. The proposed deep learning CNN algorithm has been designed for image classification and fed with post-Resistance Spot Welding workpieces images: the network has the ability to classify the images on the basis of whether the welded piece in the picture shows an expulsion or not. Despite the acquired dataset is much smaller than the ones proposed by other literature applications, the developed algorithm is surprisingly performant by giving rise to high classification accuracy values. Such an algorithm could be a sub-model to be included within an overall digital twin framework, which might be adopted by companies for a more automated quality monitoring over the outcomes of a welding assembly: expulsion appearing on a welded piece, indeed, acts as a window into the welding process and is typically considered by manufacturers as an indicator for quality of the process.

Table of Contents

Introduction

In the Industry 4.0 era, the manufacturing landscape is undergoing a deep transformation driven by the fusion between physical systems and digital technologies. The technologies enhanced by this revolution give companies the opportunity to improve efficiency and quality of their internal processes. Among the leading enabling technologies of Industry 4.0 is Digital Twin that, especially when combined with advanced Machine Learning algorithms, can provide huge potential in various industries, enhancing the strong capabilities of both technologies. On the one hand, digital twin provides a virtual replica or model of a physical object, on the other hand, machine learning consists in the development of algorithms that can learn from data and make predictions or classifications on the basis of what they learnt. A machine learning-based digital twin is also defined as 'Intelligent Digital Twin', since it acquires additional capabilities through the adoption of such algorithms with respect to a plain digital twin, in terms of prediction, optimization, advanced detection, and many more.
This thesis work embarks on a journey to explore the digital twin technology and its potentiality towards smart and efficient manufacturing processes, understanding how it allows for bridging and closing the gap between physical and digital worlds through real-time seamless data exchange between the two environments. A particular focus is given on applications within the welding processes field and on the strong potentiality of 'Intelligent Digital Twins'.

The present thesis work is structured into 6 chapters.
The first chapter provides an overview of Industry 4.0, giving a description of its main enabling technologies and of the potential benefits that the adoption of such technologies might introduce within a company, as well as the obstacles which can negatively interfere with digitalization. In addition, the main concepts and criteria behind Industry 5.0 are given, for a better understanding of what the next steps will be in terms of digitalization.
The second chapter deeply analyzes the concept of Digital Twin, describing its fundamental principles and trying to provide a general and clear definition of the technology derived from the reading of several paper works and articles. Indeed, disparate frameworks exist for digital twin and a uniform consensus about a unique definition is still missing. Then, a description of the typical main components of digital twin is given as well as an outline of the main purposes and uses for which digital twins can be adopted within an enterprise. The ISO framework proposed for digital twin is analyzed to have a better understanding of a digital twin's components. Finally, the main sectors of application of the digital twin technology are identified, the main providers of platforms needed by companies to adopt the digital twin for their processes are listed, and some general and interesting trends regarding the digital twin evolution are provided.
In the third chapter, machine learning and deep learning are described in detail, and a precise characterization of their structure and complex functioning is provided, with special focus dedicated on deep learning and Artificial Neural Networks, particularly, Convolutional Neural Networks.
The fourth chapter gives an overview of the main welding processes which have been encountered when performing the literature review about digital twin applications on welding processes. This has been instrumental for a better understanding of the kinds of digital twin applications analyzed across the different welding processes.

The fifth chapter, indeed, provides a description of the research process performed to carry out a literature review over digital twin applications to welding processes, and a systematic classification of those articles and paper works that were considered to be inherent and interesting for the purpose of this thesis work. This has been done so to have an overview of the existing digital twin applications within the welding process realm and proceed with the actual development of a practical application.
To this end, the sixth chapter proposes the development of a Convolutional Neural Network to address the problem of quality monitoring in welding processes. Such network could be adopted within a much larger digital twin framework to help with the identification of issues that might affect the overall final quality of the weld. The proposed CNN works for image classification in the context of resistance spot welding. Such CNN has been designed to analyze input images depicting assembled workpieces after the welding process has been performed, and its objective is to detect the presence or absence of expulsion defect during the welding process, which can strongly impact the quality of the assembly. To conclude, an analysis of the results obtained and a description of the possible future works, which might improve the present one, are provided.

1. Industry 4.0 overview

1.1 Industrial revolutions, from 1.0 to 4.0

From mechanization in the first industrial revolution; to assembly lines, mass production, and new power sources in the second industrial revolution; computers introduction and Programmable Logic Controllers in the third revolution; up to the fourth industrial revolution today, namely, Industry 4.0 (*Figure 1.1*).

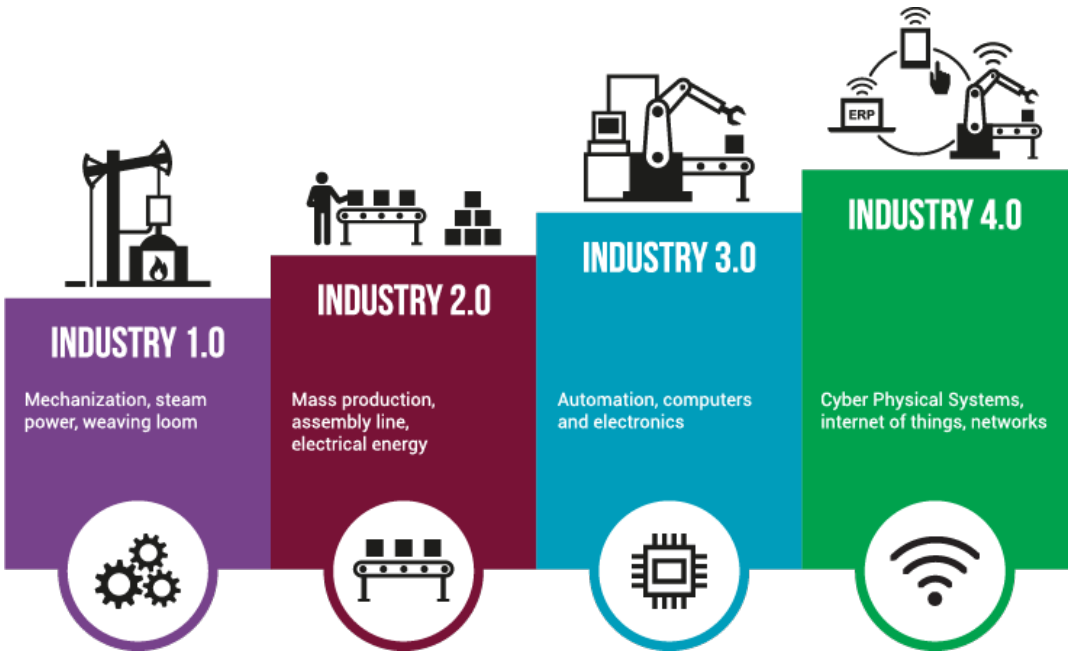

Figure 1.1 – The four industrial revolutions [1]

In the first industrial revolution, occurred between the end of 18th and the beginning of 19th centuries, machines were introduced into factories through a process of mechanization of production, substituting the much slower and more inefficient hand production. The use of water and, particularly, steam power in industrial field was a huge change and accelerated manufacturing times by improving productivity: steam engines were used to produce energy on the one hand, and to move and transport goods much faster through steam-powered locomotives on the other.

In the second industrial revolution, started at the end of the 19th century, new energy sources were introduced (oil, gas, and electric power) as well as huge technological advancements, improvements in communication technologies, assembly lines, and mass production so that production processes got much faster than before, and some level of automation was already introduced into factories.

In the third industrial revolution, begun in the middle of the 20th century, computers and robots were introduced in production processes, so that a more advanced degree of automation was reached, also thanks to improvements in telecommunications and the birth of Programmable Logic Controllers (PLC) on machineries. PLCs are industrial computer control

systems and their use in production lines was crucial to allow for a step forward in industrial automation: they receive input data (either automatically collected through sensors or manually introduced by an operator), process such input data based on some pre-defined custom application and parameters, and generate outputs which allow the PLC itself to perform different actions like monitoring, carrying out performance analysis (e.g., on the basis of productivity or other machine parameters, like temperature, etc.), signaling failures or problems of machines, and so on [2] [3].

The current Industry 4.0 revolution is implying a crucial transformation on value chains and in production, thanks to the introduction of new intelligent technologies and the consequent digitalization of the manufacturing and distribution processes, which allow for higher efficiency and productivity, also thanks to the use of informed data. Starting from the disruptive introduction of automation and computers in the third revolution, Industry 4.0 is boosting it through the adoption of smart and autonomous systems (more intelligent and flexible [4]), improvements in data and connectivity, human-machine interaction, and integration between physical objects and information world [5].
One of the outcomes of the introduction of such new technologies is retrieved in the employment of the so called 'Smart factories' or 'Cyber-physical production systems', which are intelligent and connected systems integrating advanced sensors (IoT), big data transmission, data storage and processing, artificial intelligence-enabled learning, and other software to collect and analyze data faster and more precisely than human beings can do, so to allow for better decision-making process and management. Thanks to real-time collected data through the sensors, these factories allow for intercommunication between machines (networking), self-optimization of the processes, predictive maintenance to reduce downtime to the minimum by predicting failures, and real-time visibility over the manufacturing resources (for instance, through digital twin technology) [6] [7] [8] [9] [10] [11] [12] [13].

Today, the demand for high-quality manufactured products is higher and higher, and intelligent solutions and technologies are needed in order to meet such request [5].
Consequently, more and more countries are selecting Industry 4.0 as a strategic target to develop their industrial sector, since it is proving high potential and raising lots of expectations [14], so that the global manufacturing industry is overall projected towards digitalization, networking, and intelligence [15].
Indeed, as of the second half of 2021, over a trillion U.S. dollars has been globally invested in industrial digital transformation, taking into consideration 6 main sectors (*Figure 1.2*) [16].

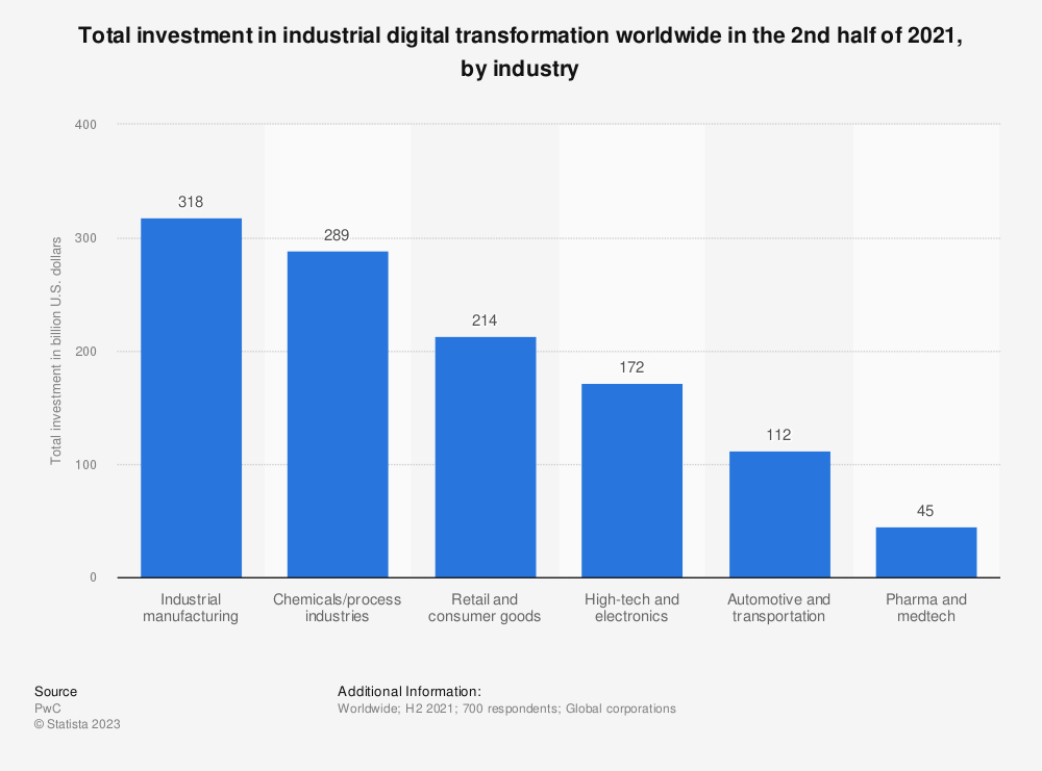

Figure 1.2 – Global investment in industrial digital transformation 2021 [16]

1.2 Main technologies

Recent technology advancements are allowing Industry 4.0 revolution to happen, enabling the digitalization of manufacturing processes and realizing the foundations needed to achieve smart manufacturing systems [17] [18].

The main Industry 4.0 enabling technologies are listed and described in the following.

- Internet of Things (IoT).

  IoT, in broad terms, refers to physical objects being enclosed in a network, communicating, and exchanging data one with the other thanks to embedded sensors, software, and other technologies like cloud computing.
  Today, any common object can be connected via IoT and when applied to the industrial sector, this technology is referred to as Industrial Internet of Things (IIoT). Plants, equipment, and machines on the production line can be 'smart', meaning that they can be equipped with sensors that gather real-time information about their current status and allow them to interconnect and communicate one with the other. Overall, this allows for higher efficiency and smoother management of supply chains, other than easiness in designing products and monitoring production lines, through collection, analysis, and exchange of a large amount of operational data, which can be reported from one asset to another one, or to an operator [6] [19] [20] [21] [22].

- Cloud computing.

  Cloud computing consists in the provision of different computing services and resources over the Internet, among which servers, software and applications, data storage, development, and networking tools. This technology provides powerful computing capability, so that sophisticated models can be built and operated with its adoption. Through cloud computing, it is possible to avoid the need for physical servers and data centers, by accessing cloud-based services simply having Internet access.
  In manufacturing, Industrial Cloud Computing provides IIoT the infrastructure needed to transfer and store data for software and applications, so that operators can use it to enhance operational productivity and assets management. A complete smart manufacturing system, indeed, requires connectivity between supply chain, production, and distribution, which is made possible thanks to cloud computing and IIoT [23] [24] [25] [26].

- Edge computing.

  Edge computing is a computing framework according to which data is stored and processed near the source (at the 'edge'), that is, near the location in which data is produced (e.g., near IoT sensing devices).
  When applied in the industrial field, it allows organizations firstly to improve response times (latency) related to the time from when data is produced to the moment in which a response on such data is needed, and secondly, to save in terms of bandwidth available. For instance, it might be needed to have real-time response and action in case there is a safety or quality issue with a machine [27] [28].

- Cybersecurity.

  Cybersecurity is the practice of protecting internet-connected critical systems (hardware and software) and sensitive data from malicious digital attacks.
  As organizations develop connectivity and communication between equipment and plants, it is crucial to apply cybersecurity to the systems, which is more and more exposed to cyber-attack threats [29] [30].

- Blockchain.

  Blockchain technology is based on a secured database that is distributed and shared in a network of members. This technology allows for safe data sharing, thanks to a ledger in which all transactions are recorded and in which participants can access up-to-date information. The blockchain network allows for valuable assets or information to be tracked as they move around the network.
  In the industrial realm, blockchain technology is a powerful tool which can be applied, for instance, over the lifecycle of the products and over the whole value chain, in order to improve transparency from raw material purchase to delivery of the final goods, by constantly monitoring the supply chain [31] [32] [33].

- Advanced data analytics.

  Advanced data analytics provide autonomous or semi-autonomous methods for data analysis which allow to extract more valuable information from big data thanks to a variety of sophisticated techniques (statistics, data mining, machine learning, predictive models, etc.) able to identify useful patterns in the raw datasets.
  These advanced models are able to forecast future patterns, other than deriving a picture of the current status of the business and the market (as traditional data analysis models already did): by applying these innovative tools, businesses are able to gain more value from their data to have support in their decision-making processes, improve their understanding of the business and the market they operate into, and how they will most probably evolve in the future.
  In factories, for instance, advanced analytics tools allow to identify bottlenecks more easily and to early detect signs of failures before they burst, so to reduce downtimes [34] [35] [36] [37] [38].

- Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (MR).

  Virtual Reality allows an individual to enter an entirely computer-generated 3D virtual environment, offering an immersive visualization experience. The user interacts through their senses with the artificial surroundings, which are close to reality, and made possible by an immersive simulation.
  Augmented Reality integrates real world with computer-generated information and elements, so to enhance real world by providing an interacting experience to the user. While AR applies to a real-world setting, enhancing both the virtual and real worlds, VR, on the other hand, implies the user to be immersed into a fully virtual environment and as such, it enhances only the virtually simulated world.
  Today, some hybrid form of Mixed Reality (MR) exists, that combines both virtual and augmented reality making real and virtual environments indistinguishable. Indeed, MR allows to reach integration between real and virtual environments and can assist in optimizing decisions through real-time data collection and scene construction [15].
  These technologies can be of help in manufacturing from different points of view: in the design phase they can help visualizing the future product behavior; in plant design it is possible to try and test different layouts via a simulated environment; a detailed visualization of manufacturing machines and lines is possible, to identify problems or improve productivity; operators can be safely trained without the need of risky physical training procedures; and many more [39] [40] [41] [42] [43] [44].

- Robotics and automation.

  Robotics is a branch of engineering and technology that involves ideation, design, construction, and use of machines (autonomous robots) whose role is to carry out physical actions usually performed by human beings.
  In line with this, automation is the introduction of technologies (software and machines) to perform activities so to reduce human intervention in processes to the minimum.

Robotics and automation intertwine in some circumstances, for instance in the case robots are used to perform some physical tasks on a manufacturing line: industrial automation and robotics, indeed, are defined as the replacement of manual human labor to improve quality, performance, and efficiency of the production line or plant [45] [46] [47] [48] [49].

- Additive manufacturing (e.g., 3D printing process).

  Additive manufacturing is a computer-controlled manufacturing process to build objects one layer at a time: 3D products are constructed by adding one layer of material upon the other, using CAD (Computer-Aided-Design) software and 3D object scanners.
  The first main benefit of additive manufacturing inclusion into manufacturing processes relates to a lower raw material wastage since, differently from traditional processes, it does not start from a 'block' of raw material and then refine it to get the outcome required, but starts from nothing and adds what is needed. In addition, it makes individual customization easier, allows for fast achievement of the first prototype and its production, and for integration of CAD software in the process [50] [51] [52] [53].

- Artificial Intelligence (AI) and Machine Learning (ML).

  Artificial Intelligence is a sub-field of computer science, in which smart machines are built to be able to solve problems and perform tasks that are usually carried out by human beings. This is possible because such machines replicate human intelligence processes by modeling and even improving them, so to provide automation of operations and business processes.
  Machine learning is a branch of AI, which focuses on algorithms that are able to learn from large amounts of data, in order to find patterns and features, and then classify objects in a given dataset or predict new data accordingly.
  Artificial Intelligence has many applications today (e.g., e-commerce, education, robotics, healthcare, etc.) and is gaining an important role in manufacturing: thanks to the huge amounts of data generated by the factory, AI software are able to identify trends that can be used to optimize plants (i) by making the production process more efficient and less energy-consuming, and (ii) by performing quality checks which allow to make any necessary adjustments to the machines.
  The application of machine learning in industrial processes, for instance, can allow for predictive maintenance based on algorithms, so to improve uptime and efficiency [54] [55] [56] [57].

- Digital twin.

  A digital twin is the virtual replica of a real object or system, and it mirrors the behavior of its physical counterpart.
  The virtual twin receives data in real time from the physical twin: by (i) integrating IoT sensors able to capture important parameters and information from the object under analysis, and (ii) applying machine learning algorithms, simulations, and advanced

analytics, the digital twin technology can enhance decision making and performance assessments. In addition, it can help operators in monitoring resources and addressing quality issues, by also providing prediction capabilities [58] [59].

1.3 Benefits of Industry 4.0 and digitalization

Increasing automation, smart factories, and availability of informed data allow for higher efficiency and productivity across the whole value chain, and higher flexibility of manufacturers in meeting the demand (e.g., through mass customization) as well as increased profitability.

More in detail, with respect to the scenario before Industry 4.0, today many advantages are faced by those companies that believe in how the current revolution could positively impact their business and implement such new technologies in their processes.
In the following, a description of some of the main benefits granted by the adoption of Industry 4.0 innovative technologies is presented [60] [61] [62] [63] [64] [65]:

- Higher productivity and efficiency. Production line throughput is higher thanks to a more efficient and cost-effective resources allocation and to the reduction of machine downtime, due to better analysis and monitoring of the line data and parameters, and more advanced predictive maintenance procedures.

- Supply chain optimization. According to the interconnectivity feature typical of Industry 4.0, the supply chain can be connected within a network: real-time data is available from the whole supply chain (information about manufacturing processes, suppliers, customers demand, inventory levels, production schedules, etc.) and this allows to optimize the satisfaction of customers' orders by improving logistics, balancing demand and supply, and upgrade the manufacturing process accordingly. To do so, the manufacturing processes can be monitored in real-time so to identify malfunctions, issues, or bottlenecks as soon as possible and intervene to improve manufacturing efficiency.

- Faster and easier product design. Thanks to technologies like digital twin and simulation software, it is possible to virtually design and test products before actually building their prototype and physically testing it. In this way, the time needed to develop a new product and start its production is much reduced, positively affecting the time-to-market. In addition, material waste is decreased, which could be substantial if many physical prototypes were to be developed and tested before reaching the best product shape.

- Optimization of products and production lines. This is mainly possible thanks to digital twin technology (combined with all the Industry 4.0 technologies embedded in each specific application), through which a virtual version of a product or a production line can be created, so to test and enhance its performance and characteristics under different possible circumstances within the virtual world.

- Instant access to data, better information sharing, and increased collaboration. Thanks to the advanced telecommunication technologies, different departments and teams within a company can easily communicate one with the other, no matter their physical location. This communication process can occur 'manually' between operators or 'automatically' between machines.

- Higher-value work. Thanks to automation, more repetitive tasks can be assigned to automated machines and autonomous robots so that human labor can focus on more value-adding jobs. Resources are overall better allocated and higher value is achieved.

- Transparency and improved decision-making. Operators within the factory or company are able to access, exchange, and analyze high-quality data generated either by sensors or by advanced algorithms as an output. Consequently, based on this crucial information and insights about the production process status and thanks to the sharing and collaboration between different teams, they can make informed and data-driven decisions. Such decisions can also be taken by intelligent machines themselves and be communicated to the operators.
  For instance, the healthiness parameters of a machine can be analyzed and showed to the operator, who can take decisions accordingly (e.g., arrange or review the predictive maintenance scheme).

- Higher flexibility and customization. Market demand and customer expectations evolve rapidly and the possibility to collect and analyze big data allows to obtain useful insights on market trends, respond fast to any fluctuations in market demand, and customize production based on requirements.

- Improved quality assurance and defect detection. Thanks to increased automation, real-time in-line quality inspection can be carried out by constantly monitoring and keeping track of the line parameters and status, so to identify any quality issue before actual failure takes place and improve the overall product quality. Also, human mistakes in quality inspection can be avoided through robotics and product compliance with quality standards can be overall improved.

- Increased personnel safety. First, the much easier information sharing and communication between employees makes crucial information, like evacuation plans or on-site safety, readily available to everyone who might need them. In addition, it allows to have standardized safety plans even across large and dispersed companies. Secondly, employees and operators training can take place virtually, so to avoid any unfortunate accident. Finally, the advanced predictive models make it possible to predict the equipment faults and possibly prevent them from taking place and causing injuries.

- Lower time-to-market. Companies adopting innovative technologies of the fourth industrial revolution have a competitive advantage due to the abovementioned benefits, since they are able to get to the market faster with their products or services. Their production activities are overall improved, cycle time is reduced while throughput is increased through better allocation of resources, and the time needed

for new product introduction (design, prototyping, and production) is much reduced. As long as the company keeps on investing in such innovations and improvements as they become available, then it is able to grow and keep such competitive advantage over time.

- Lower costs. The introduction of Industry 4.0 technologies requires an investment and so, an upfront cost in order for the factory to reach the 'Smart Factory' status. Yet, the manufacturing costs in the longer-run will decrease as a consequence of the adoption of such technologies, because of the advantages listed above and what derives from them (e.g., less machine failures and lower downtime, lower maintenance costs, better resources allocation, and less quality problems).

- Higher revenues and profitability. As a consequence of the highlighted advantages, the adoption of Industry 4.0 technologies leads to higher revenues for the company, and, consequently, to higher profitability thanks to the post-investment lower costs.

- Sustainability and circular economy. Through Industry 4.0 technologies it is possible to reduce the negative impact of industries on the environment, by focusing on reduction of wastes and on recycling, other than on designing products in a way that they are more sustainable. In addition, through collection of real-time data it is possible to keep track of the generated emissions and energy consumption so to apply some improvement strategies where needed.

1.4 Obstacles of Industry 4.0 and digitalization

Despite all the benefits granted by Industry 4.0 and digital transformation, some disadvantages and obstacles in implementing its technologies need to be taken into account [1] [64] [66] [67]:

- High upfront initial investment in capital expenditures. The introduction of new technologies is much expensive, but in case such a large investment can't be afforded by the company at once, it is possible to start with a small investment in smart technologies so to scale it up step by step as it becomes possible.

- Unclear transition process and cultural obtusity. Companies do not always recognize the utility or understand how to move towards a digitalized and automated production process. In addition to that, many companies' management is deeply risk-averse and not willing to welcome disruptive changes to their processes, and workers, as well, might not be ready for the innovation and refuse to adapt to it. Indeed, the introduction of these new digital technologies proves it necessary to draft new business models, abandoning the more traditional ones by reviewing strategies and operations.

- Lack of necessary skills and need for trained personnel. On the other hand, some companies have difficulties in finding skilled staff or in training it to the use of new software and instruments, thus slowing down the adoption of the new technologies

in many industries. Indeed, the personnel needs to be trained and specialized to be able to interface with the new technologies and automated systems.
As a consequence of this and of the previous point, a huge gap is emerging between those companies that manage to implement Industry 4.0 technologies and those which don't.

- Cyber-security and outage risks. Concurrently with the adoption of Industry 4.0 technologies, proper security systems must be introduced since data exchange and storage expose the company to hacking and data leaks risks. In the same way, network maintenance processes are needed to prevent outages to take place by disrupting business operations and production processes.

- Need for continuous update. Technology is progressing and evolving fast, and companies need to keep up by updating themselves so to remain competitive in their market.

- Increase in unemployment. Unemployment will eventually increase due to automation of processes through machines and robots which perform tasks previously carried out by human operators. At the same time, though, new occupational profiles and positions are emerging for the exploitation and management of the new technologies.

1.5 What's next? Industry 5.0, a human-centric approach

It seems like industrial revolutions are not yet come to an end. Automation and digitalization have been the focal points of Industry 4.0, but nowadays a new paradigm is emerging: the cruciality and centrality of human beings in the revolution, for a stronger cooperation between men and machines, a deeper focus on sustainability and society matters, better talent attraction, and general resilience.
This is Industry 5.0, not an actual industrial revolution, but rather a human-centric development of the Fourth Industrial Revolution, whose focus was mainly on the introduction of smart technologies along supply chains and the consequent improvement in efficiency and productivity of the systems [68] [69] [70] [71] .
According to the European Commission, the aim of this new transition is that of going beyond sole efficiency and productivity goals, towards the reinforcement of industry contribution to society, with Industry 5.0 paradigm being complementary to Industry 4.0 technologies: indeed, the wellbeing of the workers and society is central, and the general objective is to transit towards a 'sustainable, human-centric and resilient European industry' [68].
While machines and robots are characterized by efficiency, precision, and speed, they miss creativity, critical thought, and resilience which are typical of human beings, so that human-machine cooperation might allow companies to give value added to their processes.
By leveraging on Industry 4.0 technologies, it is possible to improve the interplay between humans and robots (e.g., AI and machine learning for more collaborative robots), and achieve a positive impact on society around 3 main pillars [69]:

- Sustainability goals can be reached through the (i) introduction of circular economy processes, (ii) use of renewable energy sources and recycling procedures, and (iii) reduction of energy consumption, CO_2 emissions, and wastes.
- A higher resilience and agility degree along supply chains is needed and this means being able to face any disruption which may occur, like wars or pandemics. Indeed, recent events, like COVID-19 pandemic and Russia-Ukraine war, enlightened some weaknesses in industries: higher resilience is crucial for industries to keep competitive on an international level.
- The working environment gets more interesting with Industry 5.0, since it leverages on the creativity of operators, and this makes it easier for companies to attract and retain talented and skilled workers.

2. Digital Twin (DT)

In the context of Industry 4.0 and digital transformation, Digital twin is a promising technology which is becoming increasingly relevant and proving to be a crucial concept in digitalization, by providing an innovative strategy to integrate physical and digital worlds so to produce several benefits for the organizations which are implementing it in their processes.
However, there is not yet a univocal and clear digital twin definition and framework, nor a consistent procedure to be followed, and consensus about the properties of a digital twin and its components is still missing. In this thesis work, a definition for digital twin is derived by merging information found on several articles and research papers centered around the digital twin implementation topic. In addition, a brief dissertation about the digital twin history and evolution is reported, since considered useful to identify and understand which the distinctive features characterizing this innovative technology are. Furthermore, the chapter includes a description of the main components that usually constitute a digital twin framework, an introduction to the ISO standard around digital twin adoption, and a classification of digital twins by type of application and sector.

2.1 What is a Digital Twin?

The Digital twin is one of the emerging technologies which are taking part in the Industry 4.0 digitalization process of organizations, and when adopted by companies in combination with other digitalization technologies (e.g., IoT, ML, AR, etc.), it is able to provide real benefits to the processes, for instance, in terms of lower costs, higher efficiency, reduced time-to-market, supported decision-making process, improved competitiveness, and better risk management [72].
A digital twin is a real-time virtual and computer-based replica of a physical entity (e.g., system, product, service, machine, equipment, manufacturing process, etc.) which enables sophisticated interaction between the physical object and the virtual counterpart [73]. It is a digital model that mirrors the existing physical object by collecting measurement data from the real environment: virtual models of the real objects are created, and the virtual twin is continuously updated through constant real-time data collection. Measurement data works as a sort of bridge covering the gap between physical and virtual domains and is extracted

through sensors appointed over the physical object, which are able to derive information about the real entity status and performance parameters.
Then, such data is processed and based on this information, the digital twin is able to provide several services over the whole product lifecycle:

- Constant mirroring of the real system (e.g., through 3D modeling) and visualization of its status parameters, allowing for real-time monitoring.
- Analysis of the system's parameters to produce significant insights.
- Advanced simulations of the real object actions and behaviors based on real-time acquired data, or simulation of a to-be physical object (e.g., a product or a production line) to support the design and testing phases through virtual prototyping.
- Predictions about how the real counterpart will behave, to possibly suggest optimizing improvements to be applied on the real entity, give suggestions for maintenance schemes (e.g., predictive maintenance), and grant overall support to the decision-making process.

With the abovementioned services, the digital twin is able to grant support during the whole lifecycle of the physical system, from its design to engineering, operation, and optimization stages, as long as the system keeps on being continuously updated to the real assets. One of the major advantages provided by the digital twin is the possibility to virtually carry out tests, optimizations, maintenance procedures, etc., without interrupting actual operations in the physical environment [74].

The name 'twin' derives from the ability of the digital replica to simulate not only the same elements as contained in the physical system, but also the same functioning dynamics so that it runs simultaneously with respect to the real counterpart like a twin [13].
The integration of IoT technology within the digital twin framework allows for easy extraction of real-time data and data communication, so that the digital twin can replicate what happens to the real entity and give real-time feedback to the users. Also, with the integration of cloud storage technology, large volume of data can be stored and be available to the digital twin [5] [75] [76] [77] [78] [79] [80] [81].

As said, digital twin is one of the key enabling technologies to achieve Industry 4.0 objectives and realize industrial intelligence and, as such, it is crucial to build intelligent and smart manufacturing processes and factories. Thanks to the variety of services it can provide, the digital twin has the capability to optimize the whole product lifecycle phases and intelligently support the decision-making process, to achieve data-driven optimization of the systems in a wide range of industrial applications.
This is possible thanks to the symbiotic virtual-real interaction and the existence of a bilateral communication bridge and information mapping between the physical and the virtual systems: such connection allows to cover the gap between real and virtual domains and to perform real-time analysis of the physical system.
Such real-time feature is what really is at the core of digital twin concept [14] [81] [82].

2.2 Digital Twin history

Despite digital twin technology is gaining most of its relevance in recent years only, the concept is actually quite old, and it keeps on evolving over time giving rise to a large variety

of different, at times even inaccurate, definitions and unsuccessful applications of the technology in many industries [72].

The 'twin' concept is around 50 years old, and it dates back to the creation of a mirroring replica of space vehicles implemented by NASA during the Apollo program in 1970s, which allowed to monitor the equipment conditions during the mission [83].
NASA had 15 high-fidelity spaceship simulators that were used by astronauts before departing for training purposes and were able to simulate several failure scenarios which might have occurred once the actual space vehicles were to travel in space. In Apollo 11 and Apollo 13 missions, after the spaceships took off and were traveling in space, some severe issues arose and the simulators allowed the mission controllers to prevent the catastrophic scenarios to affect the missions, and to take astronauts back home safe and sound.
Such mirroring was carried out through simulators, which by themselves cannot be considered as digital twins: what features made those simulators what today we might define as a primordial digital twin?

- Rapid adaptation and modification of simulators settings so to reflect the real-time conditions of the spacecrafts.
- Possibility to set the operating parameters of the real space vehicles, after having tested them under several possible operating scenarios.
- Connection between the physical entity and the virtual replica through 'real-time' data exchange, achieved by NASA through the telecommunication systems available at the time, by which information was received from the crew on the space vehicle and used to change simulators' settings for them to mirror the current conditions of the real vehicle.

The concept of Digital twin was then conceptualized through the 'Mirror Space Model' by Professor Michael Grieves at the University of Michigan (beginning of 2000s) [75], who proposed a Product Lifecycle Management (PLM) system to virtually map a product throughout its whole lifecycle. The proposed 3-dimensional framework included all the main elements characterizing what today is called digital twin: (i) the real physical system in the real space, (ii) the digital entity in the virtual environment, and (iii) the connection channels for the real and virtual spaces to exchange information and be synchronized. Professor Grieves defined the digital twin as a "virtual digital representation equivalent to physical products" [78] and described the model he proposed by underlying its dynamic nature, since it was supposed to be changing and evolving over the lifecycle of the physical system, keeping the real-digital connection active for this whole timespan. Initially, the real system does not exist, and a 3D-modeled virtual prototype can be developed within the digital twin virtual space, avoiding the need to develop several expensive physical prototypes to find the right shape for the system. Thanks to simulations within the virtual environment, it is possible to test the system and understand its behavior under as many as possible circumstances by varying the simulation parameters: this allows to avoid the expensive (at times destructive) tests that need to be performed on the physical prototypes in absence of a virtual twin and to prevent a much larger number of unexpected issues which might arise in the following phases of the system's lifecycle. As the final version of the object is virtually reached and validated through simulation, the 3D model is translated into its physical twin system. In the next phases, information flows over the linkage between physical and virtual systems in both ways: the virtual twin keeps on updating itself according to any change that occurs at the level

of the physical twin, and the information entering the virtual environment are used to make predictions about parameters related to the physical system future behavior.
Figure 2.1 represents the actual slide that Professor Grieves used at the time to present his model: Mirror Space Model name derives exactly from this layout, where the virtual system 'mirrors' or 'twins' the existing physical system by being its digital replica and allowing for support in design, testing, manufacturing, and monitoring phases [84].

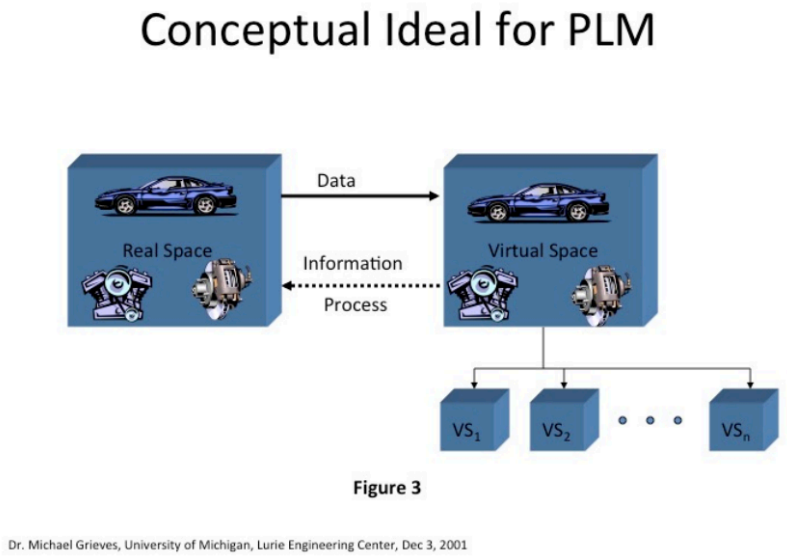

Figure 2.1 -Slide by Professor Michael Grieves [84]

In 2011, Professor Michael Grieves finally coined the system he previously proposed with the term 'digital twin'.

Despite the first real definition of digital twin was given by Professor Grieves at the beginning of 2000s, at the time technologies were limited and digital twin had no practical applications. NASA introduced the digital twin concept in its technological roadmap in 2010, defining it as 'an integrated multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its flying twin' [85] [78] [86]. The actual development of the first digital twin model was carried out by NASA itself in 2012 when it adopted integrated digital twin models to mirror the life of the space vehicles. These models allowed for design, maintenance, and predictions for the aircrafts [87].

In 2014, Michael Grieves developed a new digital twin model, still made up of three components: physical entity, digital entity, and connection between them. In this model, the virtual replica contains both geometrical data about the real system and behavioral information about its reaction to external conditions [78].

Around 2015, the advent of IoT technologies enhanced the diffusion of digital twin in the industrial field, and companies like Siemens started developing DT platforms for real-time monitoring, inspection, and maintenance as additional services for their customers [78].

In 2017, Tao and Zhang developed a five-dimensional digital twin model for the design of shop-floors composed of physical shop-floor, virtual shop-floor, shop-floor services, shop-floor digital twin data, and connections [88]. This framework has been at the basis of many applications that were afterwards developed.

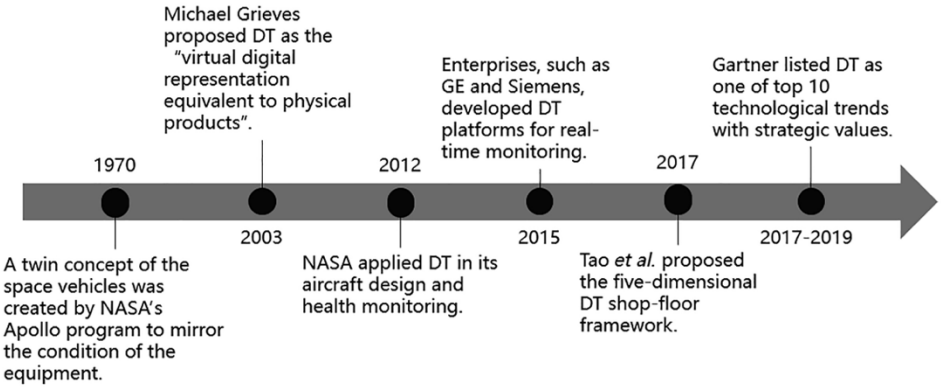

Figure 2.2 - Main steps in Digital Twin history [78]

2.3 Components of a Digital Twin

Conceptually, a digital twin is composed of 3 parts: real object in the real world, virtual representation of that, and information linking real and digital worlds [80].
To understand the digital twin structure, functioning, and main components of a digital twin, it is useful to analyze the five-dimensional framework developed by Tao and Zhang in 2017 [88], who proposed a model to reach interaction and convergence between physical and digital spaces of a shop floor. The five components of the described model are:

- Physical shop floor. It includes all the physical entities like human operators, equipment, and material.

- Virtual shop floor. It models the physical system from geometry, physics, behavior, and rule perspectives, and it synchronously evolves with the physical counterpart. Its duty is that of giving the physical system control orders, to align it with some predefined process characteristics.

- Shop floor service system. It is an integrated service platform containing functions like algorithms, models, etc., and it provides them to physical and virtual systems in response to their requests, to support the physical shop floor management and control, and the virtual shop floor operations. It also provides data services: all data collected, firstly undergoes a data conversion process to be turned into a unified form, then through data cleaning any 'dirty' data is removed, and finally, data fusion is applied to derive consistent interpretations of the data.
  Services are provided on the basis of data collected, being it sensor data (e.g., equipment capacity), simulation data (e.g., equipment fault prediction), or additional data (e.g., market data, product lifecycle data, etc.).

Abstracting from the shop floor framework, the digital twin services layer refers to what the digital twin objective is in the specific application. For instance, the digital twin can allow for monitoring, perform diagnosis and predictions of specific parameters like health conditions and remaining useful life or fault predictions, and many more. Often, the same digital twin model can offer multiple services.

- Shop floor digital twin data. It is a dataset containing data from (i) physical shop floor (generated by the physical entities, e.g., production data, environment data), (ii) virtual shop floor (e.g., model parameters and data from simulation, prediction, etc.), (iii) shop floor service system, and (iv) fused data from the three of them (which constitute more consistent and valuable data), other than (v) information about the methodologies embedded to carry out modeling, optimization, and predictions.

- Connections between all components, which allows communication and interaction between them.

Physical and virtual shop floors, and the shop floor service system keep in consistency one with the other and iteratively optimized. In particular, on the one hand the virtual shop floor is updated through real-time data about the current state, generated by the physical shop floor and collected through external sensing devices or sensors embedded within the equipment itself; on the other hand, the virtual shop floor gives back control orders to the physical space according to evaluation, simulation, prediction, and optimization models, for it to reach synchronism with the characteristics of a predefined process. Any lack of consistency between the real and virtual systems might be due to either issues in the physical entity (e.g., failure) or inaccuracies in the virtual model: in both cases, measures are to be taken to bring them back to synchronism.
The digital twin not only works with real-time data, but also records historical data generated by the system past operation (always updated with new incoming real-time data) to derive new knowledge via data mining methodologies and to improve the models' accuracy.

According to the framework, a virtual model to be complete needs to contain geometrical, physical, behavioral, and rule information about the real object:

- 3D geometric model, including information about the shape, size, position, etc. of the physical object.
  To visualize the geometrical information of the real entity, a CAD (computer-aided design) software can be adopted.
- Physical model, including information about tolerances, material properties, etc. and physical phenomena like deformation, corrosion, etc.
  An advanced design CAD software can be employed together with a Finite Element Model (FEM).
- Behavioral model, representing the way in which the digital model reacts to external inputs and stimuli, like variations in the surrounding environments and interaction with other entities. Among other solutions, FEM or neural networks can be used to represent this mapping between external inputs and behaviors.
- Rule model, which gives knowledge to the previous models so that they are able to analyze, evaluate, predict, and optimize the performance of the real entity through algorithms like data mining or neural networks.

During the physical entity operations, these models run synchronously with it and both models and physical system are calibrated to reach higher-fidelity results.

Starting from this five-dimensional digital twin model, it is possible to analyze the variety of enabling technology to be implemented in order to construct a high-fidelity digital twin model [78].
Firstly, the physical entity needs to be able to communicate data to the digital twin, since data is the media by which the digital twin can be able to understand, respond, and interact with the real world: then, sensing devices need to be adopted to acquire data and allow the virtual model to update itself and keep track of any changes taking place within the physical world.
Secondly, a large amount of heterogeneous data is generated and some advanced technologies are needed to manage data collection, data transmission, data storage, and data processing.
Clearly, for data collection, transmission, and storage the use of IoT sensing technologies, cloud computing, and communication protocols is needed. High-fidelity connection methods are needed for data transmission and to make real-time mapping possible, and such data exchange between physical and digital realms can take place through wire or wireless transmission.
Data processing consists in the extraction of valuable and meaningful data from the large set of data available, since raw data by itself is useless if not cleaned, compressed, transformed, etc.
Through big data analytics models, it is possible to (i) perform data visualization, to communicate and allow visualization of data via tables and graphical representations, to (ii) carry out data mining, through application of algorithms to find hidden information in large datasets, and to (iii) apply predictive analytics to derive real-time insights and make predictions about the future based on an analysis of historical data (statistical models, machine learning, etc.).

In the digital twin framework provided by Tao and Zhang, the model is referred to a shopfloor, but to have a general understanding of the components of a digital twin, the same 5-dimensional framework can be interpreted considering a different physical entity, which might be an individual machine, equipment, or manufacturing line within the whole shop floor.

What is crucial, but often neglected, is the importance of considering within the digital model the effect of surrounding environmental factors on the physical entity, to ensure consistency between physical and virtual environments. To do so, beside the virtual model of the physical entity of interest, a virtual environment model needs to be deployed, containing geometry, physical, and behavioral information about the environment as well [78].

2.4 A possible classification for Digital Twin models

According to a report [89] analyzing 100 existing digital twin projects, digital twin models and applications can be classified along 3 different dimensions:

a. Lifecycle phase of the real entity (from design to disposal).

The digital twin, indeed, can be applied to different object lifecycle stages like design, manufacturing, distribution, operation, maintenance, and end-of-life (e.g., recycle, disposal, etc.) [82] [78].

b. Use/purpose of the specific application.

c. Hierarchical levels.
This classification dimension depends on what the digital twin replicates within its model, which can be a component or part, an asset, a system or unit, or a process:

- The component or part digital twin, represents individual components of an entire system (e.g., motors, valves, etc.), which typically are the key ones directly affecting the overall performance and efficiency, so to keep track of their status, performance, and behavior, and improve them when needed.
- Asset or product digital twins describe entire physical assets or products (e.g., buildings, machines, etc.) and how the single components within them work together. Basically, it is a set of individual component digital twins, which allows to have an overview over the whole entity status and performance, so to make improvements if needed. For instance, by applying predictive maintenance to reduce downtimes (mean time between failures and mean time to repair) and increase the asset's efficiency.
- System or unit digital twins are the digital replica of a set of individual assets or products working together as components of a larger system (e.g., a manufacturing line): they are a combination of individual asset or product digital twins, giving the possibility to improve the collaboration between them and optimize productivity and efficiency.
- Finally, a process digital twin represents a collection of systems working together (e.g., an entire factory comprised of employees, production lines, and machines) and provides information about the interaction between all the individual units, so to make sure that each of them pursues its goal efficiently [90] [91].

In line with this tridimensional classification, the analyzed existing digital twin applications (100 works analyzed by the report [89]) can be clustered into 6 main identified groups:

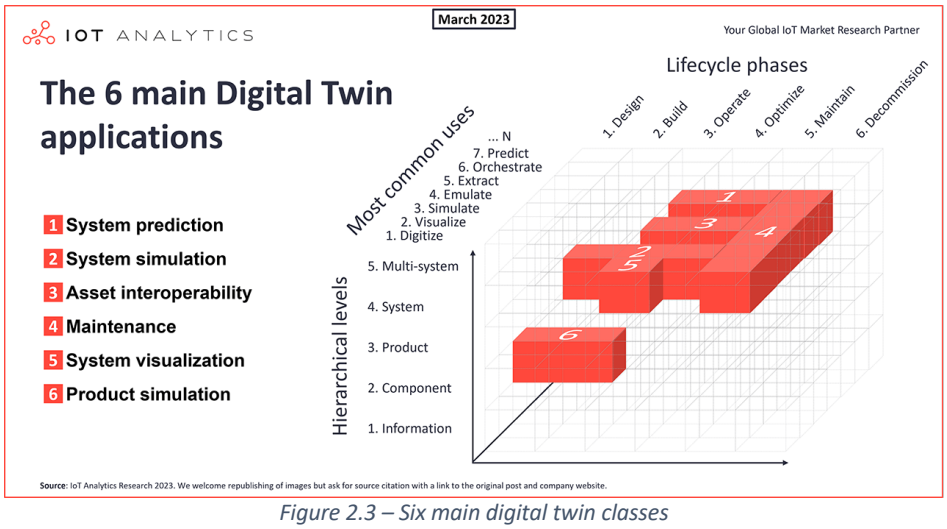

Figure 2.3 – Six main digital twin classes

1. Digital twin for system prediction. Predictive models are applied to predict the future behavior or state of an entire system or part of it (e.g., factory, building, city, etc.) during the operation and optimization phases of its lifecycle, based on current and historical data. The prediction of a theoretical future output makes it possible to improve the planning process, and with the comparison between the theoretical output derived by the model and the actual output which will arise, it is possible for the digital twin to signal the presence of any issues or anomalies in the system.

2. Digital twin for system simulation. The system is simulated under many what-if scenarios, which help in the moment of building the system itself, or apporting major changes to it (during the design, operation, and optimization phases of its lifecycle). This a priori virtual testing of the entities and their parameters allows to reduce the costs of physical testing processes.

3. Digital twin for asset interoperability. Interoperability can be defined as the capacity of two or more products or systems to communicate and understand one another's data. In line with that, such a digital twin model streamlines and standardizes the input and output data related to the asset during its operate and optimize lifecycle phases (e.g., asset features, properties, parameters, measurement data, etc.). All streamlined data are available in the same digital twin model, making it easier and faster to make more data-driven decisions.

4. Digital twin for maintenance. These models help operators in maintenance during scheduled downtime or repair by providing detailed information about the real entity. They might also provide predictive models to predict asset failures, remaining useful life and arrange for predictive maintenance, to reduce unexpected failures and downtime costs.

5. Digital twin for system visualization. The system can be visualized during its operation lifecycle phase, usually through extraction of data from sensors, 3D visualization software (e.g., CAD) and, sometimes, even machine learning algorithms to monitor the real-time operating conditions and parameters of the physical system.

6. Digital twin for product simulation. The model helps during the design and development phase of a new product or the improvement of an existing product, by simulating different possible designs before actually realizing them and their expected behavior under different scenarios (for instance, through CAD and/or CAE software and with models able to reproduce product behaviors, in terms of fluid dynamics, mechanics, electromagnetics, etc.). This allows to perform virtual tests of different design solutions and to avoid the time needed for traditional product development and testing, and the costs of constructing physical prototypes, other than improving the overall performance efficiency of the (to-be) product.

So, not one single type of digital twin exists, but it strongly depends on the specific application and the goal to be reached through the implementation. Companies should be able to quantify the benefits the digital twin may bring them in the specific use case, making a comparison with the opportunity cost of not adopting the digital twin technology. Also, they

should assess whether they are actually ready to introduce such advanced technology like the digital twin into their organizations, in terms of internal know-how, and technological and economical readiness.

2.5 ISO 23247

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies) [92]. It is a non-governmental organization composed of standards bodies (i.e., national standards organizations), each representing one member country (more than 160 countries). Each member cooperates to develop and promote international standards for a large variety of fields like technology, scientific testing processes, security policies, working conditions, business continuity, societal issues, and more. For a company, to be ISO-certified means that the business assures a service, product, or system that meets the requirements of the particular ISO standard [93].

As already mentioned, digital twin concept is evolving, and its interdisciplinary nature, combined with the existence of several different proposed application frameworks, makes it difficult for companies to understand how to introduce digital twin technology in their manufacturing processes. From this point of view, it seems that digital twin depends on the specific use case and can be defined as a fit-for-purpose approach to be adapted to the specific context of application: for instance, depending on the use case, the digital twin will have different functionalities (e.g., real-time monitoring, synchronized simulation, etc.), sensors will extract only needed data rather than all the data available from the real entity, and so on [17]. Yet, a general and adaptable application framework might be of help.

To these ends, the ISO federation intervened by proposing a standardized definition and a general framework for the digital twin implementation (ISO 23247), so that each specific organization can adapt it to their own specific use case.
The ISO 23247 series, issued in 2021, concerns a digital twin framework for manufacturing which supports the creation of a digital twin of several physical entities (called Observable Manufacturing Elements) like equipment, manufacturing processes, personnel, etc., and is structured into four parts:

1. ISO 23247-1
2. ISO 23247-2
3. ISO 23247-3
4. ISO 23247-4

According to the standard [79], a digital twin monitors an Observable Manufacturing Element, that is a physical element with material existence (i.e., a physical entity), and constantly updates data, so to enhance manufacturing operation and business cooperation. It supports anomalies detection in manufacturing process through real-time control, predictive maintenance, machine learning, etc.

The standard outlines 8 kinds of Observable Manufacturing Elements:

a. Personnel. Employees engaged in the production process, either directly or indirectly.
b. Equipment. Physical entity performing a task directly or indirectly related to the production process (e.g., robots, machines, etc.).

c. Material. Physical elements used in the manufacturing process of a product, either directly used as part of the production process or as a support material to the production.
d. Process. Sequence of physical manufacturing operations (e.g., assembly process, maintenance process, etc.).
e. Facility. Infrastructure related to the manufacturing process (e.g., production building).
f. Environment. Boundary conditions to the manufacturing process (e.g., temperature, humidity level, etc.).
g. Product. Output of the manufacturing process (either work in progress or final product).
h. Supporting document. Any artifact taking part to the manufacturing process.

In line with the standard, the digital twin of each Observable Manufacturing Element is characterized by 7 measurable and/or computable attributes:

a. Identifier. A unique value identifying the Observable Manufacturing Element.
b. Characteristic. Certain distinctive feature of an Observable Manufacturing Element (e.g., the type of equipment, the color of a product, etc.).
c. Schedule. Some temporal information about the physical entity (e.g., working hours for the personnel, the utilization time of an equipment, etc.).
d. Status. Condition of the physical object (e.g., in process, idle, etc.).
e. Location. Geographical position of the entity.
f. Report. Description of the tasks carried out by an Observable Manufacturing Element.
g. Relationship. Description of how two or more Observable Manufacturing Elements are interconnected. This is a crucial information to be able to build a database on the basis of an entity-attribute relationship structure.

Not necessarily the physical entities and the digital twin attributes defined by the standard are sufficient for all use cases: indeed, they are crucial and necessary in many use cases, yet in certain other use cases some of them might be superfluous or additional ones might need to be considered.

2.5.1 The framework

The framework outlined by the ISO 23247 [79] standard is structured into 4 layers of interconnected domains:

1. Observable Manufacturing Domain, containing the physical observable manufacturing entities.
2. Digital Twin Domain, containing the digital twins of the Observable Manufacturing Entities. It maintains information about the physical objects and their representation, and provides functionalities like simulation and analysis of data collected from the real site.
3. Device Communication Domain, containing sensors and actuators/controllers allowing the interface between digital twin and observable manufacturing domains. It collects data from the real objects through sensors, pre-processes such data, and

actuates/controls the physical entities through specific software, based on a request by the user entity or the digital twin entity.

4. User Domain, containing software applications that facilitate the users' interaction with the digital twin model and the physical entities (e.g., Product Lifecycle Management, ERP, etc.).

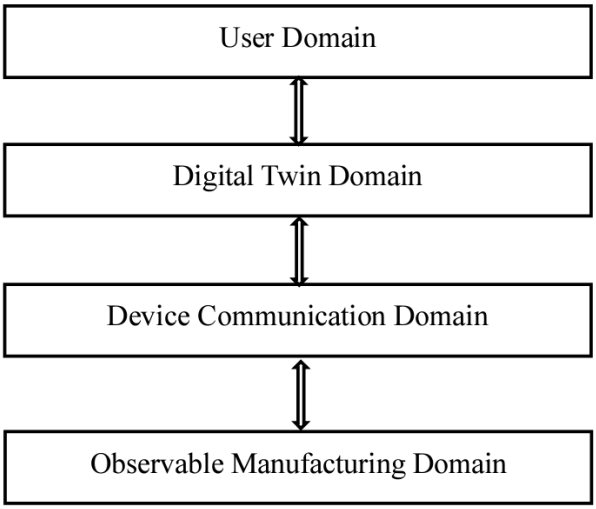

Figure 2.4 – ISO standard four layers of domains [79]

These layers are not hierarchical. Indeed, according to the framework, the whole system can run in a fully-automatic mode or semi-automatic mode: in the first case, the digital twin itself sends a command to a physical entity through the device communication layer based on some data collected by the sensors; in the second case, the user sends a command to the physical object and the digital twin updates itself accordingly, receiving data from the device communication layer.

2.6 A clarifying distinction - Digital twin vs simulation

So far, digital twin might have appeared to be no different than the existing and widely used conventional computational models and offline simulations, like Finite Element Analysis or discrete event simulation. Yet, while they both replicate products and processes through computational digital models, the digital twin shows off a key distinctive feature: it is a virtual dynamic model of the real system, with real-time data interaction through a two-way communication for synchronization between the real and virtual entities. There is a continuous connection between the real and virtual worlds, and this is what differentiates a digital twin from traditional models.

The digital twin, indeed, could embed simulation models, but carries them out based on synchronized real-time data collected by sensors from the physical objects, which makes it possible to perform better monitoring and analysis of the entities, as well as giving comprehensive and more accurate representation of the real system, and derive higher-accuracy predictions. In addition, thanks to actuators or controllers, information flows in the opposite direction as well, with the digital twin giving commands to the physical twin to maintain the two-way synchronization.

Such synchronization can be event-based, in case the digital twin is updated following the occurrence of an event, or time-based, if the digital twin is updated through a continuous data stream [79] [82].

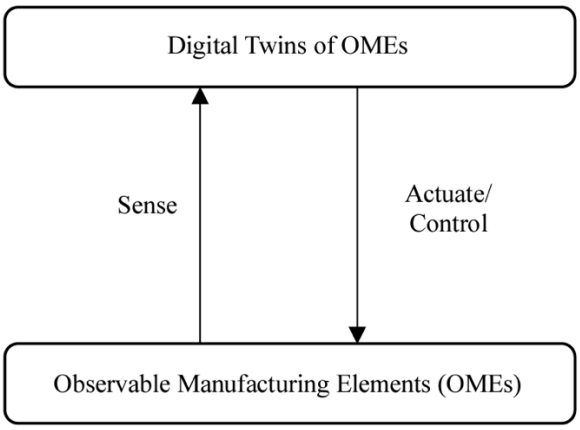

Figure 2.5 - Two-way communication for synchronization [79]

2.7 Areas of application

Digital twin technology is establishing itself in the industrial field and is revolutionizing processes along the whole value chain, creating a long-lasting competitive advantage for organizations which decide to implement it over their own products and processes. Indeed, many benefits derive from digital twin adoption, among which improvement in efficiency, minimization of the failure rates, and shorter product and manufacturing processes development time [94].
The technology, though, is not only spreading through the manufacturing sector, but also across other various industries, and some of the main digital twin end-user sectors are listed by relevance in the following [95] [96]:

1. Manufacturing. As described in the previous sections, digital twin implementation brings many benefits in manufacturing, supporting product design and prototyping, product and process optimization, quality control, and so on.
2. Aerospace & defense.
   Digital twins are used to represent and mimic functionalities of real systems like aircrafts, satellites, semiconductor subsystems, etc. This can be useful in design and R&D processes since new components can be simulated to see their performance under a set of many different scenarios; in testing, substituting physical prototyping; in performing demonstrations to customers or users; in predicting when maintenance and repairs will be needed; and many more. Overall, these services allow to improve the parts quality and even boost the lifespan of vehicles and other equipment [97].
3. Automotive & transportation.
   Digital twins are used to create the virtual replica of cars or subparts of vehicles. Benefits arise in terms of higher R&D efficiency; improved process of vehicle performance innovation and possibility to streamline the otherwise very complex car design process; carry out virtual tests; address safety issues through predictive models; etc. There are advantages deriving from the application of digital twin to manage the fleet in the transportation field, since it could provide for routes

optimization, monitoring of vehicles health, logistics improvements for delivery and shipping companies, and so on.

4. Healthcare and medicine.
   In medicine, the concept of personalized treatments is catching on, and digital twin is currently under study and development in the form of digital patient twin: a digital twin which collects in real-time current health information of the patient and is capable to predict future illnesses or monitor the patient during a treatment process, based on a biophysiological model, simulations, and large amounts of data (e.g., population studies, specifics about pathologies, etc.). Despite the technology already exists, several years are still needed for it to be introduced within real clinical applications. There is also the possibility to have support in the design of medical devices or to represent hospitals through digital twins, aiming at a better management of the facility operations (e.g., optimize patients inflow and outflow, resources allocation, structure layout, etc.) [98] [99].
5. Retail.
   Retailers can adopt digital twin technology to improve the sales and in-store customer experience through models for real-time analysis of market data and supply chains; collection of data like customer purchase trends, interests, and usual routes towards the shop; and store layout optimization. In addition, digital twin can provide for inventory management with models optimizing inventory level and predicting customers demand, as well as security optimization, creation of customer avatars, and so on.
6. Smart cities. Digital twin technology is crucial to develop smart cities, allowing to simulate plans in the urban planning and resource optimization realms, other than making it possible to even simulate the risks generated by natural disasters. In addition, by being constantly updated with the city data and status (also through IoT), it makes it possible to be always updated about the city performance and receive important insights deriving from advanced AI and analytical models. As a consequence, it might be possible to have a positive impact on the environment by having better city management, so to overall reduce carbon emissions and pollution [100].

Figure 2.6 - Main digital twin sectors [96]

2.8 Main Digital Twin providers

Given the increasing popularity of digital twin technologies, many technology providers have started to sell digital twin solutions to other companies which desire a better control over their manufacturing system and potentially improve its performance.
The digital twin offerings by 3 providers, which are considered to be among the most prominent ones, are described in the following: Siemens, Microsoft, and IBM.

2.8.1 Siemens

Siemens is a technology company which operates internationally in the fields of automation and digitalization. The company industrial business covers different segments, that are Digital Industries, smart infrastructure, smart mobility, medical technology, and digital healthcare services [101].

Siemens Digital Industries business unit is leader in industrial innovation, automation, and digitalization. It promotes digital transformation providing software and services to organizations for them to become Digital Enterprises and so, be more agile, flexible, and adaptable in developing their projects faster and more efficiently [102] [103].

The company argues that the complex challenges organizations face in their markets (e.g., more and more urgent need for sustainability) can be more easily and efficiently addressed by automating and digitalizing their processes, with a strong focus on the importance of efficiently using the generated data for optimized management of limited resources, and more informed and safe decision-making. This is exactly what a Digital Enterprise can do, by combining real and virtual worlds through collection and deep understanding of data.
According to the company, a digital twin is 'a virtual representation of a physical object and is one way to integrate real-world objects into the digital world' [104], and allows to fully exploit the potential of data availability: it collects real-time data, understands the current state of the real object by analyzing such data, and simulates its future state, allowing for optimization and earlier detection of any potential problem, also through the help of machine learning algorithms. Industrial Internet of Things clearly plays a role in their proposed approach and allows to join different processes in a single constant data flow over the whole value chain. Other state-of-the-art technologies, like AI, edge computing, cloud computing, industrial 5G, blockchain, cybersecurity, and additive manufacturing are included within their technologies for a more intelligent data exploitation [102].

The company proposes a digital twin approach to cover the whole lifecycle of the real assets (i.e., design, production, operation, maintenance), which allows companies to (i) achieve better product designs and optimize production systems faster (before the need to invest in real physical prototypes), other than (ii) allowing to simulate the operation of real assets (e.g., machines or even entire factories), and (iii) help operators to virtually practice their tasks [105].
Accurate modeling of the product and production process lifecycle is obtained through sensors placed on the physical entities, enabling for real-time monitoring of their performance and current status. The digital twin constantly evolves and updates itself to keep track of any real changes over time, by reflecting the physical entity over its whole lifecycle.

The proposed digital twin is composed of data from the physical objects, but its core component is typically a simulation model that allows for the analysis of the physical twin behavior. An analysis of the 'what if' scenarios and the prediction of future performance under different conditions through the digital twin is carried out, and makes it possible to obtain real product and production process working exactly as they were designed [106]. Such simulation model might be combined with machine learning algorithms or virtual and augmented reality technologies for an even higher performance [107] [108].

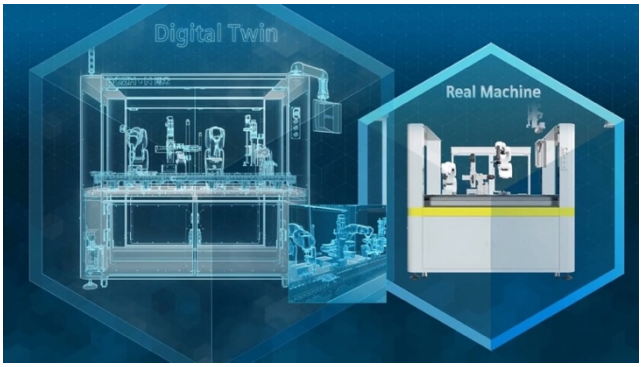

Figure 2.7 - Digital twin by Siemens [109]

According to Siemens, there exist 3 typologies of digital twins [94] [108]:

1. Product digital twin, used to efficiently design new products by virtually simulating and validating their properties and performance, according to the specific product requirements. This connection between virtual and real product allows to test the product behavior under different conditions and scenario and change the virtual product features accordingly.
   The advantages from the use of a product digital twin lie in a much faster design and in a better quality of the final product, also avoiding the need for many physical prototypes for testing.

2. Production digital twin, used to plan the production process by validating its efficacy and efficiency before actually starting production. By combining product and production digital twins, companies can make useful predictions about the future activity of the system, like predicting when preventive maintenance will be needed, being able to optimize production in advance. Potential errors and failures, indeed, can be predicted in advance and prevented before starting the actual operation process, so to save time and even allow for customized mass production, since production can be tested and predicted with derisory cost, effort, and time.

3. Performance digital twin, used to collect, analyze, and exploit operational data. Such digital twin is constantly fed with operational data collected from products and manufacturing processes, and analyzes it to provide useful information. This digital twin allows for constant monitoring of the real object current status parameters and gives support for an informed decision-making aiming at process optimization. It is also possible to perform predictive maintenance so to prevent downtime and optimize energy consumption and resources efficiency.

To develop digital twins, companies need powerful software systems and Siemens proposes its Xcelerator platform, that is comprised of a marketplace to find ideas and products, a portfolio of hardware and software solutions (PLM, CAD, simulation, etc.), and the description of some use cases proving the real benefits brought by digitalization into production processes, in terms of improved performance and higher attention to sustainability. This system uses the most innovative solutions and technologies (e.g., AI, IoT, digital twin, and Metaverse) to allow for an increasingly strong integration between real and virtual worlds. Its purpose is to support organizations (industrial, infrastructure, and mobility sectors) in their digital transformation process and value creation, by making it faster and easier [110] [111].

2.8.2 IBM

As one of the largest technology companies and strongest technological innovator in the world, IBM provides a digital twin solution with the IBM Maximo® Application suite. It is an integrated cloud-based platform providing a set of applications to carry out monitoring, management, and predictive maintenance operations, among many others more. The suite makes use of Artificial Intelligence, IoT, and advanced analytics to gain benefits like reduction of downtime and costs, and performance optimization [112].

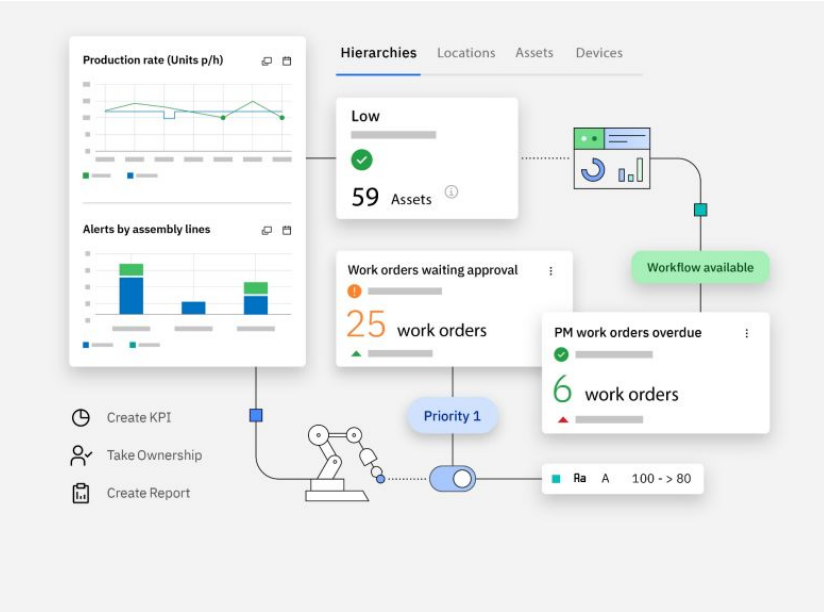

Figure 2.8 - Snapshot of the IBM Maximo® Application Suite [112]

2.8.3 Microsoft

Microsoft Azure is a cloud computing platform by Microsoft, which offers access to cloud services and applications, among which Microsoft Azure Digital Twins. It is a platform which makes use of IoT special intelligence to create and manage digital models of real-world physical environments, and to analyze, control, automate, and simulate the behavior of connected devices and systems. By offering different APIs and quick pre-built templates and sample models, the platform gives a better understanding about the systems it represents and makes it possible to make more informed decisions through the digital twin models. The

platform is highly scalable, providing for the development of several digital twin models to be applied for large-scale IoT projects like smart cities or buildings [113] [114].

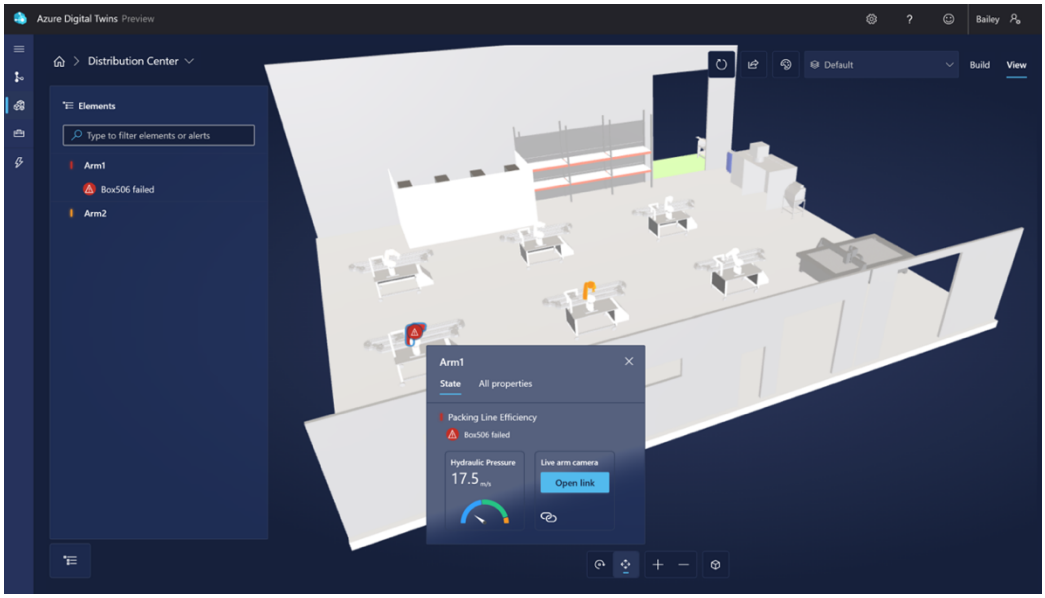

Figure 2.9 – 3D visualization of an environment digital twin [115]

2.9 Recent and expected trends about Digital Twin

Technology is advancing fast, and organizations are making great efforts to keep their competitive advantage by introducing digitalization into their manufacturing systems. Consequently, the demand by companies for the implementation of digital twin solutions is undergoing a recent increasingly strong acceleration.

The substantial growth in the digital twin market is due to different drivers, mostly related to the evolutions concerning technologies used for the creation of a digital twin (i.e., IoT, AI, cloud, etc.) [116] [80]:

- Increasing IoT adoption by companies. For the digital twin to mirror properties, behavior, and interactions of the corresponding physical entity, sensing devices are needed to collect and exchange real-time data from the real environment to be analyzed for monitoring and by prediction models.
- Advancements in big data analytics, cloud, and artificial intelligence and machine learning technologies. These technologies, when implemented within the digital twin framework, make the digital representation more powerful and 'intelligent', meaning that, for instance, it will be able to make predictions about the future of the system under analysis.
- Increasing adoption of 3D printing. This production process has issues in terms of the 3D printable materials, which may undergo some distortions during the 3D printing process. This calls for the need to perform long and expensive trial and error sessions, having a negative impact on the cost of printing parts and on the time needed to print them. To this end, digital twin helps by performing 3D simulation of the printing production process and implementing models to predict what and when potential

distortions may arise, so to prevent them by adjusting and optimizing the 3D model of the process.

- Increasing stress on the importance of cost reduction, optimization of scarce resources allocation, improvement of operational efficiency, and overall improvement of the supply chain management. The possibility to simulate several potential scenarios and make data-driven decisions, makes it possible for companies to optimize their processes, reduce risks, be more responsive and agile, and consequently reduce costs.
- Growing emphasis on sustainability. Customers demand more and more for sustainable services and products, and digital twin technology is able to ensure alignment with sustainability matters through energy consumption monitoring and optimization, environmentally oriented decision-making process, analysis and forecasting of environmental impact of projects, etc. [117].

According to a Gartner survey (2019) [118], 75% of organizations worldwide implementing IoT in 2019, were already using digital twins in their manufacturing processes or were planning to within a year: more into the details, 13% of companies implementing IoT was already exploiting digital twins, while 62% were working to implement digital twin or planning to do so within a year.
With respect to the past, digital twins are 'entering mainstream use', and this steep growth in adoption is due to the high business value that digital twins have been proving to bring.
International experts [80] (from companies in different industries, universities, and research institutes) expect the global role of digital twin in 2030 to be a nearly omnipresent one, and almost all technical products will be delivered with embedded digital twins. Factories and machines will achieve much better performance in terms of cost efficiency, productivity, and sustainability, and digital twins will become commodities and standard solutions to perform test and validate functionality ahead of time to solve future problems. Product lifecycle Management (PLM) will be much easier, since digital twin will allow to transfer information from one lifecycle stage to the other (e.g., from the design phase to the production phase). Supply Chain Management (SCM) will be more efficient as well, allowing for a quicker reaction to any potential issue (e.g., supply bottlenecks or changes in demand). Also, experts predict strong development and increasing importance in digital twin solutions combining simulation and artificial intelligence technologies [105].

In terms of sectors in which digital twin is mainly implemented, automotive and transportations markets are expected to be the ones increasing digital twin adoption the most between 2023 and 2030: indeed, they are adopting 3D printing and 3D simulation more and more to design and produce vehicles, and as said above, this goes hand in hand with an increasing need for digital twin technology.
Also, digital twin in the automotive industry allows vehicle designers to keep track of large datasets from the vehicle, so that future improvements and cost optimization is possible.
In addition, future automobiles are going to be more and more connected and autonomous, making the adoption of digital technologies crucial to pursue such purpose [80]: digital twins of cars are under development, from their design phase up to the operation phase when the vehicle is delivered to the customer (e.g., Tesla [119]).

Following, aerospace and defense, retail, healthcare, energy and utilities, IT, and telecom, as well as other secondary industries, are expected to grow at strong rate due to the increasing automation level in such sectors [96].

The results of some 2023 surveys, revealed that 63% of manufacturing companies globally are currently developing a digital twin strategy or have already done so, while 29% have fully implemented the digital twin strategy on part of their operational resources or are in the process of doing so [89].

Digital twin requires for technologies like IoT sensors, big data, cloud, AI, and many others: the inclusion of devices like IoT sensors and instruments like cloud platforms, though, is currently raising some concerns about increased risk of security and data protection. Some hacker, indeed, might steal sensitive information about the company, placing the organization's operations at risk [80]. To secure the digital twins, cybersecurity principles and best practices should be followed: a proper risk management strategy should be adopted, to identify the right assets to be protected, evaluate the impact of potential threats and apply mitigation actions; implementation of security controls over multiple layers, so that if a control fails, the threat won't get past the other control layers; since digital twins are based on data exchange, network security (e.g., firewalls, unauthorized intrusion detection systems, encryption protocols, etc.) is crucial; regular security assessments are to be performed so to identify vulnerabilities, if any; and so on [120].

3. Machine Learning (ML) & Deep Learning

3.1 What is machine learning?

Machine learning is the practice of using algorithms that analyze data, learn from the data, and then make a classification or prediction about new data [121].
The ability of such algorithms to learn from data is what distinguishes ML algorithms from traditional algorithms, which directly instruct computers to perform some tasks by feeding them with an explicit and manually written set of instructions.
In case of machine learning algorithms, the machine is trained through large amounts of data and by learning features from that dataset, it is able to perform some tasks without being told how to do them exactly.

3.1.1 The machine learning process

To adopt and implement a machine learning solution, a specific pipeline needs to be followed to go from the problem to the solution [122] [121]:

1. Identify the (business) problem to be solved (e.g., arrange for predictive maintenance, perform energy efficiency optimization, cost-saving, quality monitoring and control, etc.).

2. Identify which type of data is relevant for the specific problem to be solved, and if no dataset is available yet, perform collection of the necessary data to carry out the model training process.
   It is important that the data is of quality to obtain a final good quality model, and that the dataset size is sufficiently large for the model to be well trained over the features of the input data.

   Any dataset used to train a ML model is composed of the same entities:
   - Features or independent variables: known variables which will be used to predict the dependent variable.
   - Dependent variable or target (also referred to as 'label' in case of a classification model), which the model will be able to predict.

   So, the dataset is composed of the matrix of features (x) and the dependent variable vector (y).
   For simplicity of explanation, for now the dataset can be considered as a table containing some independent variables (x) and a dependent variable (y), like in the example below, where 'profit' represents the dependent variable, and the other columns are the independent variables (i.e., the features).

Profit	R&D Spend	Admin	Marketing	City
192,261.83	165,349.20	136,897.80	471,784.10	Rome
191,792.06	162,597.70	151,377.59	443,898.53	Milan
182,901.99	144,372.41	118,671.85	383,199.62	Turin
166,187.94	142,107.34	91,391.77	366,168.42	Florence
191,050.39	153,441.51	101,145.55	407,934.54	Naples

Table 3.1 - Example of dataset with dependent variable and independent variables

Dependent variable vector
Matrix of features

   The dependent variable is the one to predict, while the independent variable(s) or predictors are given and known, and are used to derive the value of the dependent variable.

3. Take care of missing data.

   The dataset might present some missing data which needs to be handled, since it could cause errors in the training process. In principle, imaging the dataset as a table, the observation with missing data might be ignored by simply deleting the corresponding row, but this can be done only in case of large datasets, for which the learning quality of the model won't be affected. In all the other cases, the missing value is usually replaced with the average of all the values in its column.

4. Encode categorical data [123].

   In the dataset there might be some literal labels (e.g., dependent variables taking values like yes/no, cat/dog, etc.) or categorical independent variables (e.g., cities, colors, etc.), which contain label values rather than numeric values: such strings need to be converted into numbers through an encoding procedure for the ML model to be able to derive correlations between the features and the dependent variable. Indeed, since ML models are usually not able to interpret label data and categorical variables as they are, all inputs and outputs need to be converted into numerical form, i.e., integer values or vectors of integers.
   There are 2 possible approaches to encode categorical data:
   - Integer encoding. Each categorical label or variable is converted into an integer value (e.g., 'New York' is 1, 'Rome' is 2, etc.). In some cases, this approach is enough, but it is a risky one since the ML model might interpret the existence of a numerical order between the categories, while there actually isn't any sequential relationship between them, ending up with wrong resulting predictions and poor accuracy.
   - The typical approach to perform encoding is called One Hot Encoding, which converts the categorical labels and variables into vectors containing 0s and 1s and avoids the risk of the model misinterpreting ordinal correlations between the features or labels where no ordinal relationship exists.
     Each categorical variable or label is turned into a so-called one-hot encoded vector of binary variables (said dummy variables), of length equal to the number of existing categories: for instance, if the model is supposed to label the input data as 'pizza', 'pasta', or 'meat', each of these categories will be turned into vectors containing 3 integers like 'pizza' corresponding to [1,0,0], 'pasta' to [0,1,0] and 'meat' to [0,0,1].
     Within the one-hot encoded vector, each single index corresponds to one specific category, meaning that each element in the vector will correspond to a 0 value, except for the element in the index position corresponding to the specific category: in the example above, the first element in the vector corresponds to the 'pizza' category and so it corresponds to a 1 value, while the other elements in the vector are equal to 0.

5. Split the dataset into training, validation, and test sets.

   Typically, the observations in the original dataset (e.g., rows in the table) are split into 3 subsets: training, validation, and test sets [124].

   The training set is used to train the model over several iterations, in which it understands the features within the dataset and the correlations between the variables. The validation set, instead, is smaller than and separate from the training set and the model has no information about such dataset: it is used to validate the model by evaluating its performance on new observations during the training process, perform hyperparameters tuning accordingly, and identify overfitting if any. Indeed, after the model is built and trained over the training set, it is applied to perform predictions over the new observations contained in the validation dataset. Such

predicted values are then compared to the actual target corresponding to the validation data points, in order to evaluate the model functioning.
The definition of a validation set allows to identify the best model configuration in terms of hyperparameters setting (as it will be clarified in the following sections) and to identify and avoid overfitting of the model to the training set, so that the model does not get used to perform predictions over the training set by being unable to do the same over a set of new observations, that were not used to train it. In case accuracy of the model on the training set is promising, while that on the validation set are worse, then the model is most probably overfitting.
In addition to training and validation sets, an additional separate set can be defined (the test set), to test the ML model after the training and validation process is concluded by making it perform predictions over the outputs of the test set. Usually, differently from training and validation sets, the test set is fed to the model as unlabeled: in this case, indeed, the objective is to run the model as it will be run on real applications and so, by feeding it with unlabeled data and receiving the prediction about such labels as an output. This helps in getting an overall idea of the final performance of the model, after hyperparameters tuning is concluded and the model is trained, and in understanding how different kinds of machine-learning models perform when applied to the same problem, so to choose the best one.

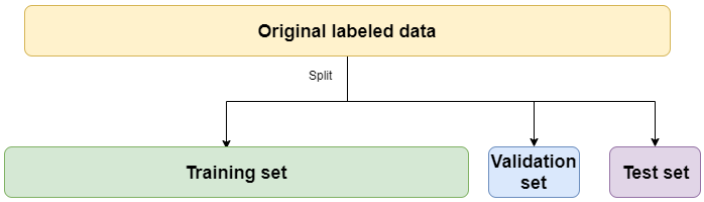

Figure 3.1 – Training, validation and test sets splitting [124]

The typical proportion is 80-90% of initial data into training set, and the rest equally split between test and validation sets; still, this splitting proportion depends on the specific application and may differ from the usual one.
What is important is that all 3 sets are representative of the population distribution, meaning that they need to contain various samples representing all the features in the population of the specific problem.

6. Perform data augmentation.

According to data augmentation, new data samples are created by transforming the existing observations in the training set.
For example, if the dataset contained images, this operation would imply to create new augmented ones by applying geometrical transformations to the images in the training set like rotating, flipping, or cutting. This would improve the capability of the algorithm to identify the categories of objects in the images even if they are not always appearing in the same positions as in the original training set samples, but are, for instance, rotated or cropped.
This is mainly done to prevent overfitting: data augmentation allows to increase the size of the training set, by creating more samples to train the model on and avoiding it to overlearn and be overtrained on the existing data samples.

Data augmentation is not applied on the validation and test datasets, which must remain intact (apart from feature scaling) because these samples need to be considered as new observations to understand how the model would perform when adopted in a real application with real data.

7. Apply feature scaling.

   One last important step in data preprocessing is feature scaling. It consists in scaling all the features (variables), to make sure that they all take values within the same scale and so are comparable. Feature scaling can be performed through normalization (obtaining values within [0,1] interval) or standardization (obtaining values within [-3,+3] range). It is not to be applied to dummy variables (but only to numerical values), since they are already included within the [0,1] range and feature scaling might cost losing the categorical interpretation of these variables.

   - Normalization: $X' = \frac{X - X_{min}}{X_{MAX} - X_{min}}$

     X_{min}: minimum value in the column
     X_{MAX}: maximum value in the column

   - Standardization: $X' = \frac{X - \mu}{\sigma}$

     μ: mean value of the column
     σ: standard deviation of the column

   It is of greatest importance to apply feature scaling after splitting the dataset into training, validation, and test sets, and not before, in order to avoid any information leakage on the validation and test sets, whose observations needs to be unknown and new to the ML model as performance validation is carried out after training. Indeed, feature scaling gets max and min, or average and standard deviation of the features in the dataset in order to perform the scaling, and if this is done on the complete dataset before splitting, it would cause the ML model to get information about variables of the validation and test set which are not supposed to be known. The best practice is to derive min and max, or average and standard deviation from the training set alone, apply feature scaling to the training set, and then blindly apply it to validation and test sets as well, but using min and max, or mean and standard deviation information derived from the training set only.
   Standardization is the most recommended feature scaling approach, since it works in all cases, while normalization works in case the features follow a normal distribution.

8. Modeling.

   At this point, the model needs to be built and trained, and the prediction to be made. A machine learning model is the computer program which is trained to recognize patterns and features within the dataset and be able to pursue a specific objective, which might be that of performing a classification or making a prediction.

Before starting the training process of the algorithm, the hyperparameters must be decided and set [125]: the value of these parameters controls the algorithm learning process and the final produced model, by affecting the parameters which the model by itself is going to learn during the training process. They are set and cannot be modified during the training process and are not part of the final resulting model, while the learnt parameters are. The goal is that of choosing the optimal hyperparameters configuration which allows the model to learn the optimal parameters, so to correctly map independent variables (i.e., features) to the dependent variable (i.e., label or target).
Some examples might be the train-test sets split ratio, or the number of hidden layers, the gradient descent learning rate, the number of hidden neurons or the number of epochs in a neural network, etc.
Parameters, instead, are internal components of the model since they are learnt by the algorithm during the training process, in which the program tries to find the mapping relationship between features and targets. They are usually initialized to some random values and updated through an optimization algorithm as the learning phase progresses. At the end of the process, they will characterize and be part of the machine learning model itself.
Some examples might be the weights in regression models and weights in neural networks.
So, it is clear how crucial the choice of hyperparameters is, since in the end it is going to affect the final parameters of the resulting model. Indeed, an important step in modeling is the hyperparameters tuning, which is the process of setting different hyperparameters configurations, and evaluating the consequent model performance over the validation set in terms of prediction or classification accuracy. There is not a rigid approach in determining the right hyperparameters values, rather a more trial-and-error approach to optimize the model performance resulting from the validation set with different hyperparameters settings, until the best one is selected.

There are different possible machine learning models, depending on the specific problem to be addressed. It is possible to use the test set so to compare the performance of different models and choose the most appropriate for the problem of interest, which might be ML regression, ML classification, or deep learning algorithms:

a. Machine Learning Regression, that is a technique through which the algorithm is trained to understand and learn about the relationship between independent variables (features) and a dependent variable (outcome). A mapping function is estimated, and the model will then be used to predict continuous real (numerical) outcomes based on the input features [126] [127] [128] [129].

    - Simple linear regression: estimation of the relationship between one single independent variable (X_1) and a dependent variable (y) through a sloped straight line.

$$\hat{y} = b_0 + b_1 X_1$$

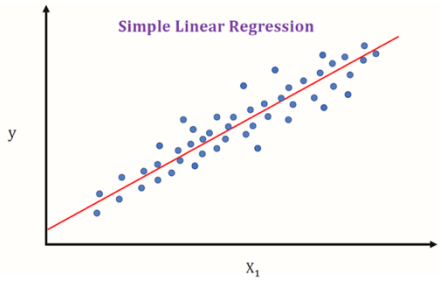

Figure 3.2 – Simple linear regression equation [129]

- Multiple linear regression: estimation of a linear relationship between two or more independent variables and a dependent variable.

$$\hat{y} = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n$$

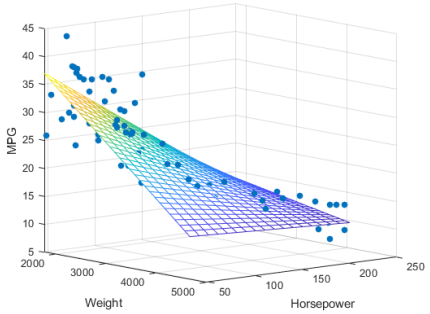

Figure 3.3 - Multiple linear regression equation [130]

- Polynomial regression: estimation of a non-linear relationship between independent variables and dependent variable.

$$\hat{y} = b_0 + b_1 X_1 + b_2 {X_1}^2 + \cdots + b_n {X_1}^n$$

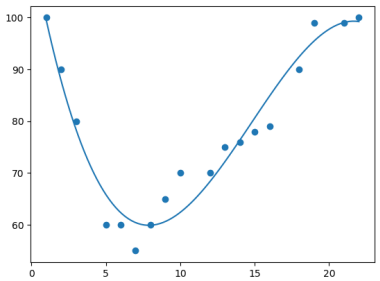

Figure 3.4 - Polynomial regression equation [131]

- Support vector for regression (SVR): estimation of a hyperplane that best fits the data points in a continuous multidimensional space, that is, a hyperplane which contains the maximum number of data points, with a threshold value for the maximum distance that can exist between the hyperplane and a data point not belonging to the plane.

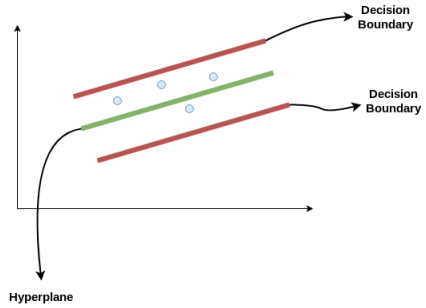

Figure 3.5 - Support vector hyperplane [132]

- Decision tree regression: the training dataset is split down into smaller subsets using a binary decision tree which is incrementally developed and contains decision nodes and leaf nodes.
  A decision node has two or more branches and splits the datapoints according to a condition over the datapoints features. After the decision tree is developed, a new data point will go through the decision tree, going to the branches of which its features satisfy the conditions, until a leaf node with the decision is reached.

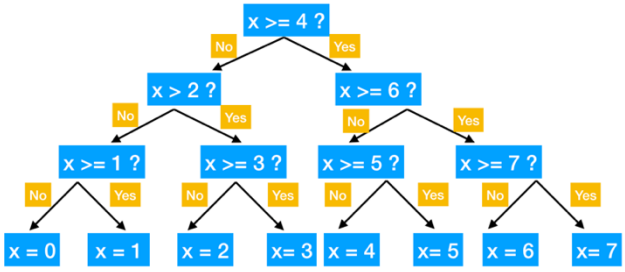

Figure 3.6 - Regression decision tree example [133]

- Random forest regression: use of ensemble learning method, according to which the predictions from multiple machine learning algorithms are combined to get a more accurate resulting prediction than the one from a single model. In particular, several decision tree algorithms run in parallel during training process and use the average of their prediction to derive the output.

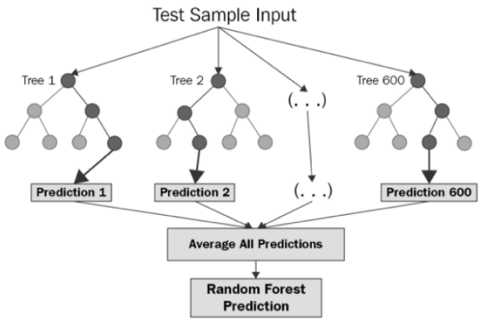

Figure 3.7 - Random forest regression [134]

b. Machine Learning Classification, a technique through which the algorithm is trained to predict and identify the category (class label) to classify an input datapoint [135] [136] [137] [138] [139] [140].

- Logistic regression: special regression model in which the dependent variable to be predicted is categorical (e.g., YES/NO). The model uses a logistic function (for which only output values within [0,1] are possible) and predicts the probability of occurrence of one specific output category, which is indeed included within the [0,1] range.

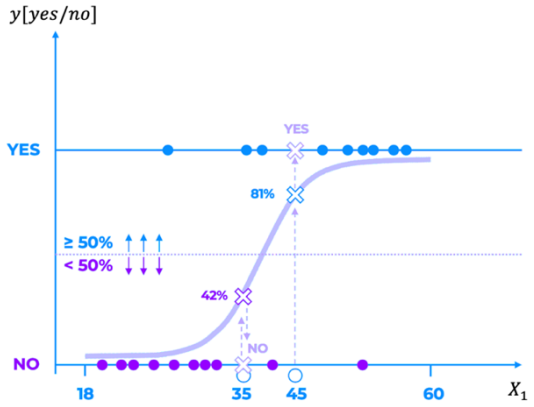

Figure 3.8 - Example of logistic regression function [121]

Considering, for example, a simple case in which there is one single independent variable (X_1), a new observation to be classified is projected onto the logistic regression curve to obtain the probability for the dependent variable to belong to a specific class (e.g., YES). A probability can be fixed as threshold above which to consider a new observation as belonging to the YES category (usually $p \geq 50\%$), so that probabilities can be turned into actual predictions (e.g., YES or NO).

- K-nearest neighbors (K-NN): algorithm whose functioning is based on the assumption that similar datapoints are close to each other. To decide the class of a new datapoint, a sort of voting procedure takes place, where the K closest neighbor datapoints to the new one 'vote' by saying the class they belong to. The new datapoint will be assigned to the category voted by the majority of the K neighbors.

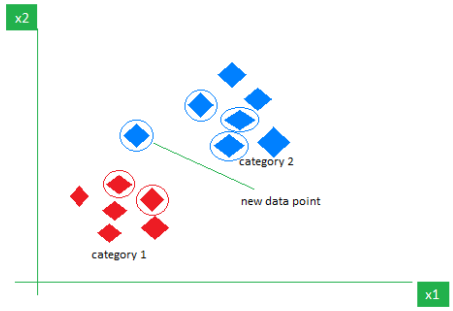

Figure 3.9 - K-nearest neighbors scheme [141]

- Support Vector Machine (SVM): estimation of a line or hyperplane in a multidimensional space that distinctly separates the datapoints into different classes, that is, the plane with maximum distance (margin) from the closest training datapoints of both categories. Based on this distinction, the model will then classify a new data point depending on whether it lies on one side of the line/hyperplane or the other.

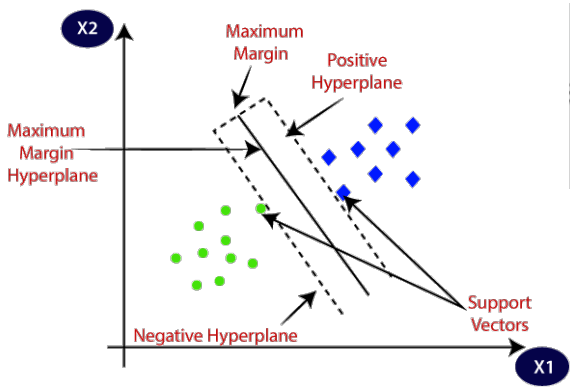

Figure 3.10 - Support Vector Machine hyperplane [142]

- Kernel SVM: Kernel functions can be applied to SVM algorithms to manipulate the data, particularly, to transform the training set in a way that an otherwise 2-dimensional non-linear decision hyperplane is transformed to a linear plane in space with a higher number of dimensions.

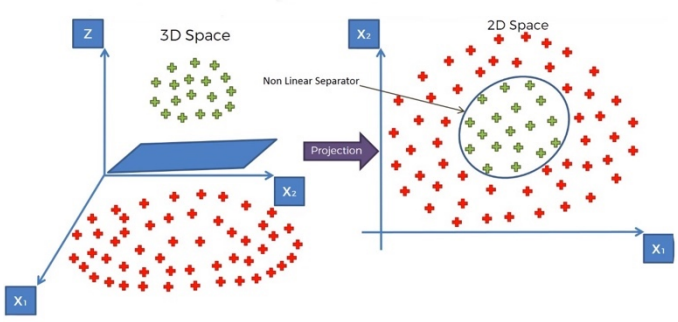

Figure 3.11 - Support Vector Machine with Kernel mechanism [143]

- Naïve Bayes classifier: probabilistic machine learning model based on the Bayes statistical theorem. The datapoints are labeled within a certain class based on their features, by applying the Bayes formula.

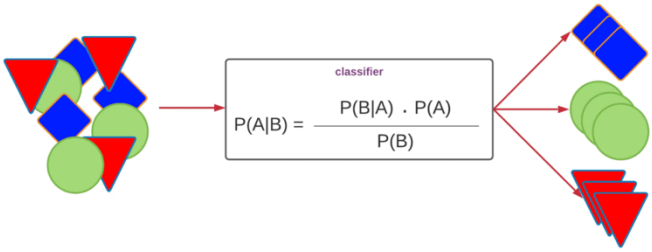

Figure 3.12 - Naïve Bayes functioning scheme [144]

For instance, considering some independent variables (e.g., 'sunny', 'rainy', etc.) and two possible labels (e.g., 'YES go out' or 'NO go out'), then if the probability of 'YES go out' given 'sunny' event is higher than the probability of 'NO go out' given 'sunny' event, then the output classification for 'sunny' input value is 'YES go out' [144].

$$P(YES\ go\ out \mid Sunny) > P(NO\ go\ out \mid Sunny)$$

- Decision tree classification: it works in the same way as decision tree regression models, breaking down the datasets into subsets to derive the output class to assign to the input datapoint.

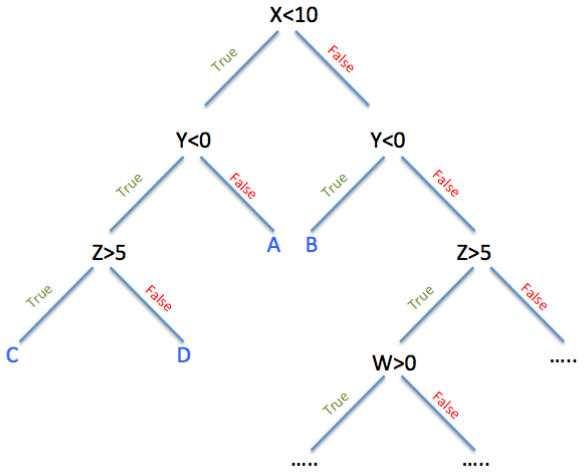

Figure 3.13 - Decision tree classification example [145]

- Random forest classification: again, it works as random forest regression models, but the final prediction is a class (label).

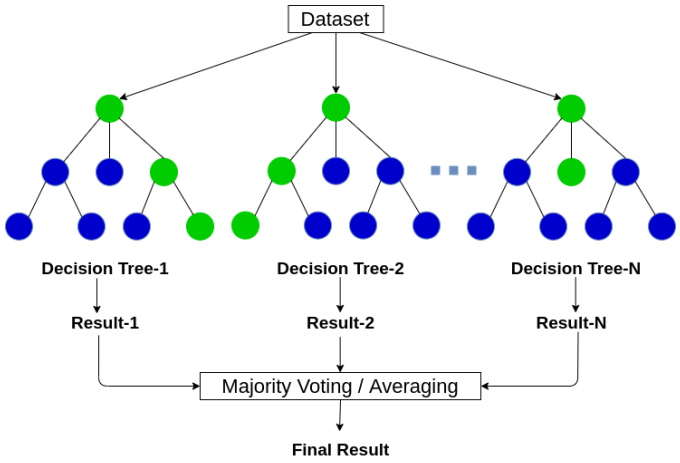

Figure 3.14 - Random forest classification scheme [146]

c. Deep learning: artificial neural networks

9. Model evaluation.

Some performance metrics are calculated so to understand the goodness of the model and to make sure it actually serves the purpose it's designed for. Evaluation is derived both with respect to the validation set (to carry out hyperparameters tuning) and to the test set (to derive the final optimized model's performance over new unseen data).
Here some of the typical performance metrics are listed:

- R-squared and adjusted R-squared, to evaluate the goodness of fit of a regression model [147] .

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}}$$

with:

$$SS_{RES} = \sum_{i\_1}^{n}(y_i - \hat{y}_i)^2 \text{ and } SS_{TOT} = \sum_{i\_1}^{n}(y_i - y_{AVG})^2$$
$$y_i = actual\ value\ of\ a\ sample$$
$$\hat{y}_i = predicted\ value\ of\ the\ sample$$
$$y_{AVG} = average\ of\ actual\ y\ values\ in\ the\ dataset$$

SS_{RES} is the residual sum of squares and indicates the amount of error between the regression function and the actual datapoints: the smaller it is, the more the regression function is well-fit to the data [148].
SS_{TOT} is the dispersion of the observed samples around the dataset mean, it measures the total variability of a dataset.

R^2 is a statistical measure that indicates the proportion of variance in the dependent variable that can be explained by the independent variables, that is, it shows how good the regression model fits the data. Its value is included within the 0-1 range, since usually $SS_{RES} < SS_{TOT}$ (i.e., the distance of datapoints from the regression curve is generally shorter than their distances from the average line): indeed, the regression curve is built so to minimize SS_{RES}.

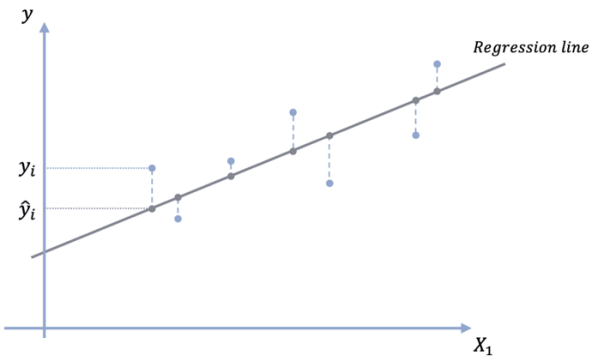

Figure 3.15 - Regression line vs predicted and actual values [121]

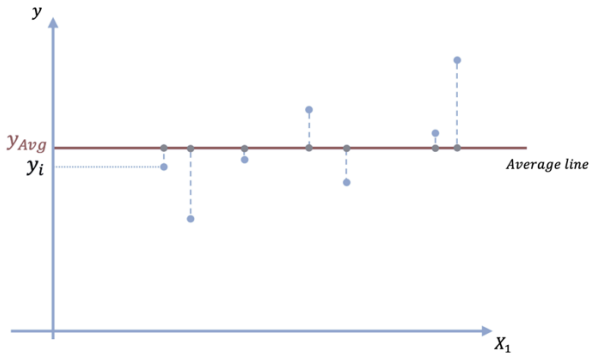

Figure 3.16 - Average line vs predicted and actual values [121]

The better the model fits the data, the smaller SS_{RES} and, consequently, the greater R^2:

R^2	Goodness of fit (rule of thumb)
1	Perfect fit ($SS_{RES} = 0$)
≈ 0,9	Very good model
< 0,7	Not great model
< 0,4	Very bad model
0	The model does not make sense for this data

Table 3.2 - Goodness of fit based on R² value

R-squared, though, suffers from a major flaw [149]: no matter the number of independent variables which are added to the regression model, its value will never decrease. This might cause the addition of an indefinite number of variables which increase R-squared while adding no actual value to the regression model (i.e., might be variables which have nothing to do with the specific problem).
As a consequence, Adjusted R-squared metric is typically used rather than R-squared, since it is better able to accurately view the correlation between one variable and another:

$$Adj\ R^2 = 1 - (1 - R^2) \times \frac{n - 1}{n - k - 1}$$

where k is the number of independent variables in the model and n is the sample size. With this new formula, when a new independent variable is added, k increases so that $Adj\ R^2$ decreases: it is like penalizing the addition of variables so that it is worth adding an extra variable only in case R^2 increases enough to compensate for the introduced penalty.

- Accuracy: ratio of number of correct predictions over the total number of input data samples predicted.

$$Accuracy\ = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ made}$$

- Confusion matrix: it shows the number of correct predictions and the number of incorrect predictions.

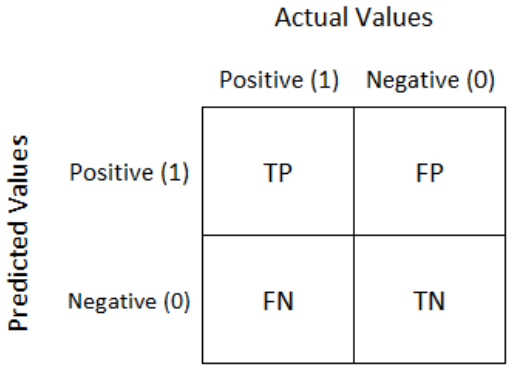

Figure 3.17 - 2x2 confusion matrix [150]

For simplicity, the confusion matrix shown as an example is related to a binary classification problem, where two possible classes can be assigned to the observations.
TP = true positives. Correct predictions of observations belonging to class 1 (i.e., observations actually belonging to class 1 were correctly predicted as belonging to 1).
FP = false positives. Incorrect predictions of observations belonging to class 0 (i.e., observations actually belonging to class 0 were incorrectly predicted as belonging to 1). This is also defined as type-1 error, according to which the model predicted an effect that in reality is not there.
FN = false negatives. Incorrect predictions of observations belonging to class 1 (i.e., observations actually belonging to class 1 were incorrectly predicted as belonging to class 0). This is also defined as type-2 error, according to which the model predicted no effect while it was actually there.
TN = true negatives. Correct predictions of observations belonging to class 0 (i.e., observations actually belonging to class 0 were correctly predicted as belonging to class 0).

From this, an accuracy rate and error rate can be defined as follows:

$$AR = \frac{Correct\ predictions}{Total\ predictions} = \frac{TN + TP}{TOTAL}$$

$$ER = \frac{Incorrect\ predictions}{Total\ predictions} = \frac{FP + FN}{TOTAL}$$

10. Model deployment and monitoring.

Once the model is ready, it can be finally introduced for real applications, for instance in production processes. At this stage, it is important to monitor the model performance as it is employed so that, if needed, it can be retrained and updated accordingly, re-evaluated, and deployed again.

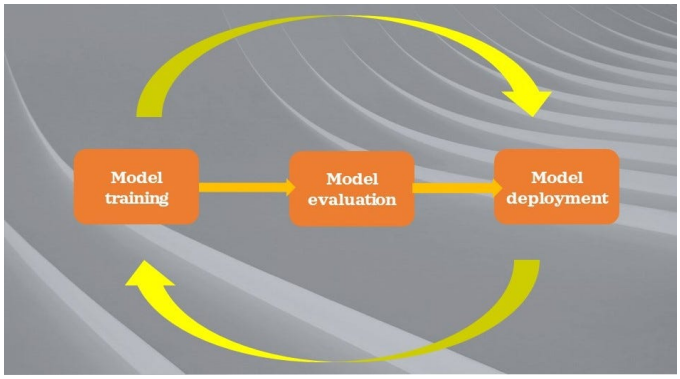

Figure 3.18 - Model deployment loop [122]

3.1.2 Overfitting issue

Overfitting occurs when the ML model becomes high performing in making predictions on the training set, but not so successful when it comes to the validation set, which contains data that were not used to train the model itself.
It is possible to understand whether a model is overfitting the training dataset by comparing the accuracy of predictions on the training set and the validation set: in case the latter's metric is much worse than the former's one (i.e., the model is much better fitting to the training set than to the validation set), then the ML model is most probably overfitting.
In addition, a model that is overfitting is not good at making predictions on the test set either, with respect to how good it is in making predictions on the training set, and this is another wake-up call for identifying the overfitting issue.

The reason why overfitting occurs, is that the model has learnt too well the features of the data included within the training set, but it is not able to generalize what it learnt so that it is not able to work on datasets which are different from the training set (e.g., validation and test sets) [122] [151] [152].

To try to overcome the overfitting issue, some measure can be taken [153]:

a. When possible, train the model on a large training set.
   Having more data available, the model will be able to learn more information from the training set. Also, a higher diversity of features is included in a larger training set, so that the model becomes more capable of recognizing features and making predictions on datasets it was not trained on. Indeed, when the training dataset is too small, it does not contain enough data samples to learn about all the possible kinds of input data values.

b. Introduce hold-out, by splitting the whole dataset available into training, validation, and test sets, so that not all the data available is used for training. This will make the model perform well on the validation and test set as well, other than on the training set, having good generalization capabilities since validation and test sets contain new observations that the model has never seen and didn't use for the training.
   The typical split is 80% for training and 20% for validation and testing.

c. Apply data augmentation, which consists in generating new augmented data by creating modified copies of the starting training set data, so to artificially increase the amount of data to train the model on in case no additional data samples are otherwise available.

d. Decrease the model complexity by changing some parameters (e.g., removing layers or decreasing the number of neurons in the layers of a neural network) in a way that the model is more capable to generalize what it learnt to new input data. An over-complex model might more likely overfit, since the model will start learning about the noise within the training data, and the right complexity balance should be found between underfitting and overfitting risks.

e. Apply dropout [154]. Dropout is applied to neural networks and randomly sets a fraction of neurons to a zero value at each training step, so that a portion of the nodes does not take part to that step of the training process. Some nodes, then, are ignored by the network and won't take part in the prediction process on the training. In this way, learning interdependency among neurons is reduced and adaptation of the model to the training dataset can be avoided so that the network gets better able to generalize over new datasets. Neurons are dropped out only during the training process, while are activated back as predictions are made over the validation and test sets.

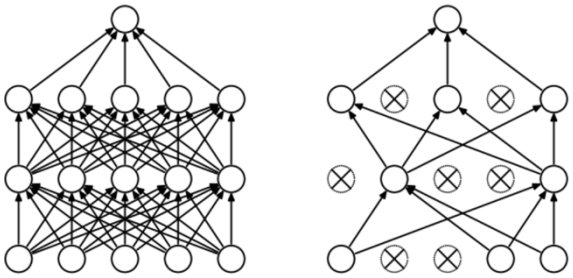

Figure 3.19 - Drop-out method [155]

f. Early stopping. It is possible to plot a graph representing the loss of the model over the validation set and see that after a certain number of training iterations (i.e., epochs), this loss starts increasing rather than decreasing. At this point, the training can be early stopped, and the current model saved.

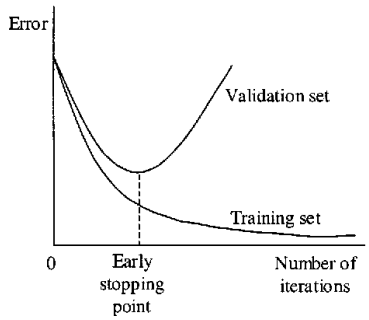

Figure 3.20 - Validation loss curve [156]

g. Apply K-fold cross-validation method. By firstly setting apart the test set, the remaining dataset can be split into K groups (folds). The training process is structured into iterations: at each iteration, one of the groups acts as the validation set, while all the others together as the training set. This process repeats itself until each group has been used once as validation set. Differently from simple hold-out methodology, this is computationally more expensive, but allows all the data available (apart from the test set) to be used for training and once for validating.
Thanks to cross-validation, it is possible to select the final model configuration: hyperparameters in the network can be tuned by using datapoints in the original training set (by extracting a different validation set at each iteration), so to keep the test set as truly unseen from the model [157].
Such method allows to address the overfitting issue, since it makes it possible to assess the model's performance over different subsets of data and so, to obtain a more realistic evaluation of the model's ability to generalize to new datasets. Indeed, with simple hold-out, much depends on the kind of datapoints included within the single validation set that is selected for the evaluation: if this is particularly 'difficult' to classify or predict, the model might achieve bad validation accuracy; however, in such case it would not be due to overfitting to the training set, but simply to the difficulty of the specifically selected validation set.

Figure 3.21 - K-fold cross-validation scheme [158]

3.1.3 Underfitting issue

Underfitting occurs when the ML model is not capable of making predictions on the training set it was trained on and, consequently, on new datasets (e.g., validation and test sets) that it has never observed. The model is not complex enough to be capable of accurately capturing and determining a meaningful relationship and mapping between input and output values (i.e., features and target variables).
When the model is underfitting, the accuracy metrics resulting from the training data are unsatisfactory, as well as from the validation and testing sets due to high error rates on all the datasets [159] [121].

Some actions can be taken to face the underfitting issue:

a. If possible, increase the number of features included within the training set so that the model can learn additional information from the dataset and be able to make better predictions on the training set, by improving its accuracy.
b. Increase the model complexity by changing some parameters (e.g., adding layers or increasing the number of neurons in the layers of a neural network). Indeed, if the model were to be too simple and the dataset too complex, then it might be that the ML model is not sophisticated enough to perform predictions on such dataset.

c. Increase the training time, making it possible for the model to learn more features from the training dataset and be able to reach higher accuracies on validation and test sets as well.

d. Reduce dropout percentage. Since dropout is applied only during the training process on the training set and not when the model is making predictions on the validation set, if the accuracy of predictions on the validation set is higher than the one on the training set, it would seem reasonable to reduce the percentage of nodes to dropout.

So, while an overfit model gives accurate results for the training set but not for the validation and test sets, underfit models give inaccurate results for all datasets (i.e., training, validation, and test sets). The right balance must be achieved to obtain a well-fit model.

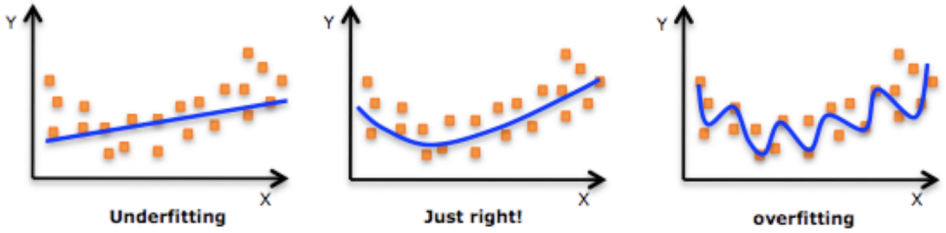

Figure 3.22 - Underfit, well-fit, and overfit models accuracies over the training set compared [160]

3.2 What is deep learning?

Deep learning is a branch of machine learning which uses particular algorithms, called Artificial Neural Networks (ANN), whose functioning is inspired by the structure of the neural networks in human brains.
Indeed, the purpose of deep learning is trying to mimic how the human brain behaves through multi-layers neural networks, in a process of learning and acquiring skills from a large amount of data, and then applying such skills to perform some actions like recognition and classification of elements within the dataset.
In particular, Artificial Neural Networks (ANN) are used for regression and classification models, and Convolutional Neural Networks (CNN) to carry out computer vision, with the overall objective of making predictions with high accuracy.

The learning process will be either supervised or unsupervised: the learning process is supervised if the model learns from an already labeled dataset, while unsupervised learning occurs if the model is fed with unlabeled data. A more detailed distinction between the two learning approaches will be given in the following paragraphs.

Deep learning is boosting automation by providing technologies able to perform tasks without the need for human intervention, so that operations get completed faster [121].

3.2.1 A useful distinction: artificial intelligence vs machine learning vs deep learning vs neural networks

Before getting into the details of artificial and convolutional neural networks, it is important to clear out and define the boundaries between artificial intelligence, machine learning, deep learning, and neural networks, which are often confused and named interchangeably [161] [162] [163].

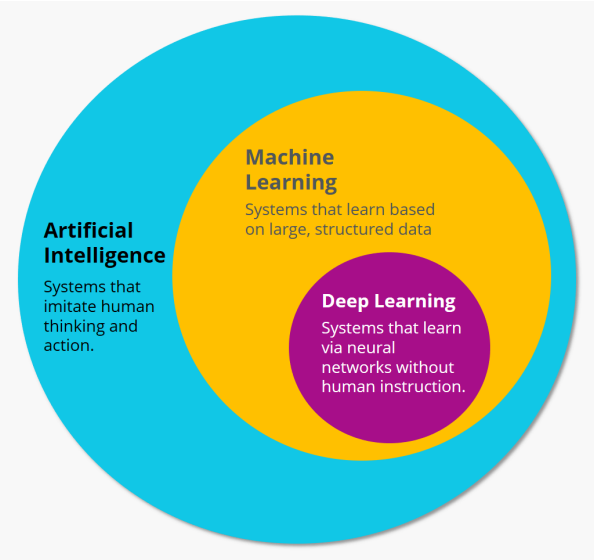

Figure 3.23 - AI vs ML vs deep learning [162]

Artificial intelligence (AI) is related to intelligence which belongs to a computer program running on a machine, and not to the human brain. A machine is considered to be artificially intelligent if it has the capability to reproduce human intelligence, by perceiving its environment and acting in it accordingly so to achieve a specific objective in problem-solving and learning procedures. Through artificial intelligence, machines are able to make predictions and to execute tasks and decision-making processes that are usually performed by humans, by even optimizing their execution and completion.

Machine learning is a subset of AI, and deep learning is in turn a subset of both AI and machine learning. They both implement AI through learning algorithms.

Machine learning uses algorithms to analyze data and learn from it: based on what the algorithm has learnt from the dataset, it will be able to infer something about new data or to derive new data (e.g., classify data or make predictions), and help in making informed decisions which minimize usual decisional mistakes.
What differentiates a machine learning algorithm with respect to a traditional one, is that they learn how to carry out tasks directly from data through a training process while in the latter case, code is developed containing a set of explicit instructions for the machine to execute it.

Deep learning, as said, is a subfield of machine learning which works through multi-layer algorithms, that is, artificial neural networks, which are able to learn and make intelligent decisions autonomously.
Indeed, plain machine learning algorithms still need human intervention to identify patterns, learn and perform actions, receive feedback on their correct and incorrect classifications or predictions and optimize the accuracy of their results. On the other hand, deep learning algorithms are able to autonomously understand, through the neural network, whether their own classification or prediction is more or less accurate.
Their functioning, indeed, is modeled on how human intelligence works and allows to reduce manual human intervention as much as possible. As a consequence, these systems are also able to work on unstructured data: this means that not necessarily training data into the network need to be previously categorized and labeled according to some distinctive features (i.e., supervised learning), since the system itself is able to automate the feature extraction procedure, by understanding the distinctive characteristics of the data and classifying it into different categories without the need for manually defined labels (i.e., unsupervised learning).
However, to be able to perform these tasks and obtain accurate and reliable results, a deep learning algorithm, working according to an unsupervised approach, needs a much larger input dataset than a general supervised machine learning algorithm, which can instead receive less data points while counting on an explicitly given data structure.

Finally, the main technical difference between simple neural networks and deep learning algorithms is in the 'depth' of node layers: to be considered as a deep learning algorithm, a neural network needs to have more than one hidden layer.

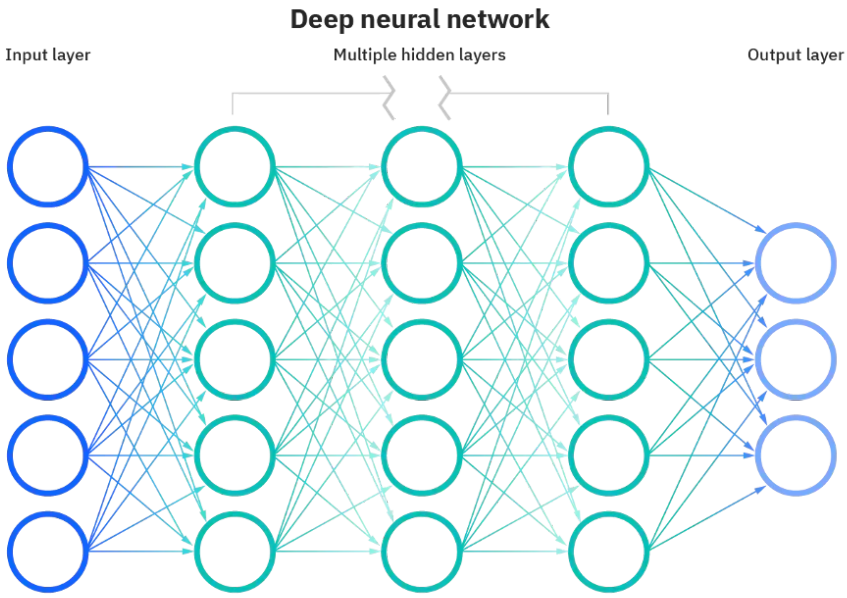

Figure 3.24 - Example of deep artificial neural network [161]

3.2.2 Supervised vs unsupervised learning

As anticipated, there are two different approaches in machine learning: the learning process of an algorithm can be supervised or unsupervised [122] [165].

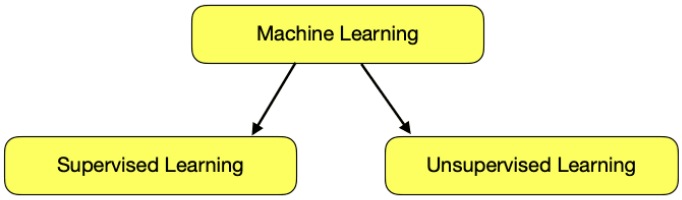

Figure 3.25 - Supervised and unsupervised machine learning [166]

In case of supervised learning, the model is trained on a previously labeled training data and validated on a labeled validation set, so to help and 'supervise' the prediction or classification process. The model, indeed, is fed with pairs containing the data sample and the corresponding label defining its output value: (sample, label). In this way, the model is able to learn from the labeled training dataset how to perform a mapping from inputs to outputs (from data sample, to output label), thanks to the labels it is provided with.
Supervised learning can take place into two methodologies depending on the problem to be solved [164]:

- Regression models (linear or non-linear) are used to predict continuous numerical variables, by understanding the relationship between one dependent variable and one or more independent variables given as datapoints. Regression algorithms can be used to perform tasks like predictive analytics (e.g., estimate life expectancy, market forecasting, weather forecasting, etc.) [167].
- Classification models are used to predict a category: based on a training procedure, the algorithm is able to identify the category of new observations (e.g., image recognition) by predicting a categorical dependent variable (e.g., yes/no in case of a binary outcome) based on a certain number of independent variables. Classification algorithms can be used for tasks like image classification.

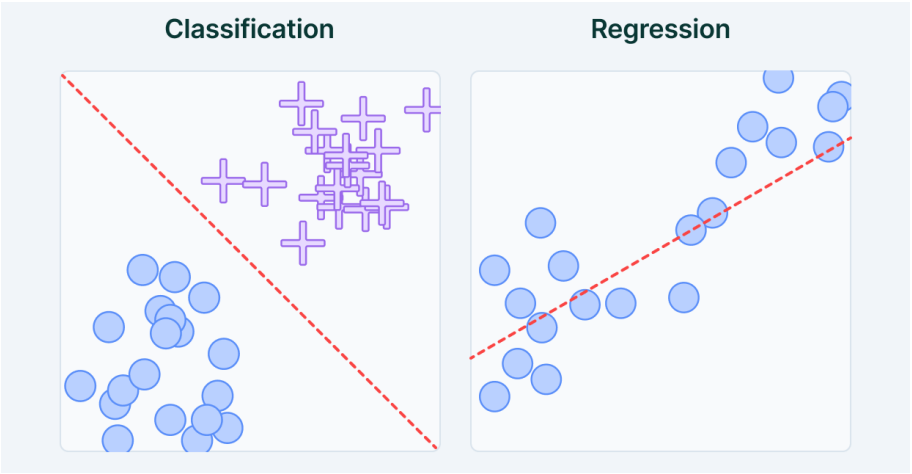

Figure 3.26 - Classification vs regression models [164]

The algorithm receives a training dataset as input and learns how to make predictions or classify new observations through an iterative procedure, adjusting itself until the right outcome is reached.

In case of unsupervised learning the provided dataset is unlabeled and unstructured, meaning that the algorithm simply receives unlabeled pieces of data (samples) with no corresponding labels and with no explicit instructions about what to do with the data. Then, the algorithm autonomously analyzes and clusters the data in a self-learning procedure, being able to identify hidden patterns, features, and structures in the dataset and map inputs into specific outputs, without the need for any human intervention.
Consequently, accuracy metrics cannot be used to evaluate an unsupervised learning model. The main areas in which unsupervised learning technique is applied are listed in the following [164]:

a. Clustering algorithms.
   The model learns the structure of the data even if it is not given any label, finds hidden patterns based on similarities or differences, and clusters the data samples into sub-groups creating clusters of data items. Data objects showing most similarities are grouped within the same cluster, while objects lacking that commonality are categorized within another group.
b. Association algorithms.
   The model finds the relationship of one data item to another one, based on the probability of co-occurrence of different items into a collection (i.e., it determines what set of data items occur together in the dataset). These dependency mappings can be used to perform marketing tasks related, for instance, to customer habits and behavior analysis.
c. Dimensionality reduction (autoencoders algorithms).
   An autoencoder is an artificial neural network which learns how to compress and encode data and to reconstruct it back from the reduced and encoded version: it takes in an input dataset and as an output it reconstructs that input with a representation that is as close as possible to the original one. For the model to have high accuracy, indeed, the reconstruction of the initial input (i.e., the output), needs to be as similar as possible to the initial data sample itself. An autoencoder, then, reduces the size of a dataset, learning how to identify and ignore the noise included within the dataset itself.
   One possible application of autoencoders is the denoising process applied on images: the algorithm is able to extract the meaningful features from highly-noised images and reconstruct them excluding the noise elements [168].

In some applications, a semi-supervised learning approach can be used, as a middle way between supervised and unsupervised learning techniques. Indeed, in this case the training dataset is composed of both labeled and unlabeled data samples.
Within the semi-supervised learning realm, pseudo-labeling technique is typically implemented. Supposing to have a dataset containing a portion of labeled samples, and another subgroup of unlabeled data: according to pseudo-labeling, the labeled portion of the dataset is firstly used as training dataset to train the model according to a plain supervised learning approach; after this first training process, the model is used to make predictions about labels of the unlabeled portion of the dataset and these predictions are used to

'pseudo'-label the unlabeled data samples; after that, the model is re-trained on the whole dataset containing both the labeled and pseudo-labeled datapoints. Thanks to this technique it is possible to train the model on a much larger dataset to get higher accuracy and to avoid the expensive and time-consuming process of manually labeling the entire dataset.

In general, supervised learning algorithms are more accurate than unsupervised ones, but they require higher degree of human intervention in structuring the data inputs. Still, also unsupervised learning models need for a slight human intervention for validation, just to be sure that the model works properly.

Here, a simple example is reported to understand the difference between supervised and unsupervised learning. Supposing to have a set of images containing pictures of 2 categories (A and B) and to feed the model with such dataset. In case the model was supervised, each image would have been previously labeled as A or B, and the model will be trained to learn about the differences between the images belonging to the two different categories. In case the model was unsupervised, images would not be manually labeled, and the algorithm would autonomously learn the features from images and classify them into A and B categories according to their differences.

3.3 Artificial Neural Networks (ANN)

Artificial Neural Networks are a subfield of machine learning and are the models used in deep learning: they are learning algorithms that receive some data as input and processes those data inputs over a series of neural layers, so to deliver an output.
The architecture of an ANN is inspired by and tries to emulate a biological brain neural network, and indeed, the basic building block of an ANN is the 'neuron': the functioning of human brain neurons is replicated in the network, where a set of different processing units (called 'artificial neurons', simply 'neurons', or 'nodes') are interconnected through weighted links and organized into layers. These neurons interact one with the other by exchanging signals over the links connecting them (synapses) and process such signals, with the objective of deriving a final output in the last layer of nodes.
Thanks to this structure, the ANN receives input data and is able to map it into some pre-defined output values through the data processing task carried out by each node: indeed, the single node generally receives data as input from the previous layer via the weighted links, performs some operations on the data, and sends such processed output as input to the next layer through the weighted connections.

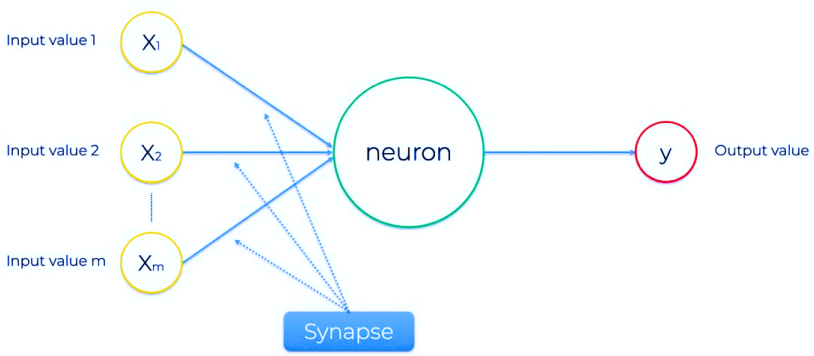

Figure 3.27 - Scheme of a neuron [121]

3.3.1 Structure

As said, neurons are organized into layers and each layer performs some specific processing operations and transformations on the input data it receives. The basic structure of ANN is composed of 3 kinds of neurons layers: input layer, hidden layer(s), and output layer.
After the input data is given to the initial layer (input layer), the signal is transmitted and processed through the network's internal layer(s) (hidden layer(s)), until it reaches the final layer (output layer), as explained in detail in the following:

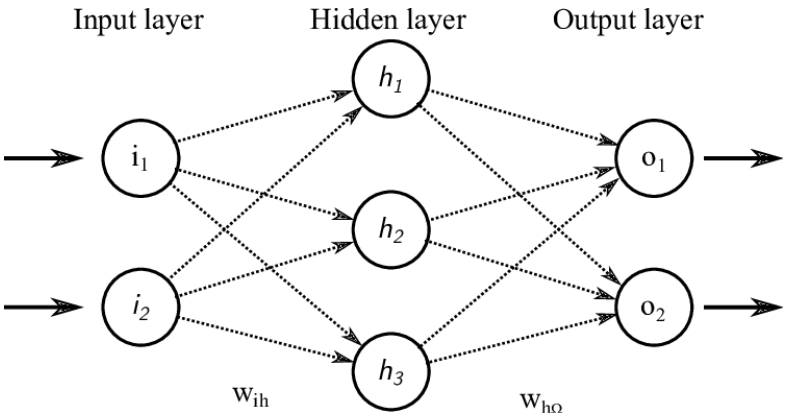

Figure 3.28 - Example of a simple ANN with 1 hidden layer [169]

1. Input layer.
   First layer of the network: a set of neurons which receives the input data, that are the known values (independent variables) injected into the model under construction, from which the network will predict some output value. At this stage, no calculations are performed yet.
   The number of neurons in the input layer corresponds to the number of features (i.e., independent variables for the specific problem) related to a single observation in the input dataset fed to the network.

2. Hidden layer(s).
   Layer(s) included between input and output layers: all computations on the input features are performed at this level, and the result is then passed on to the output layer.
   In deep learning algorithms, there are multiple hidden layers of neurons, which build upon the previous layer and optimize the prediction or classification, reaching higher accuracy of the result with respect to single layer-networks.
   So, the neural network is considered 'deep' in case there is more than one hidden layer, hence the name 'deep' learning.
   The number of hidden layers and nodes in each hidden layer can be arbitrarily chosen, based on prior experience with well-performing networks applied to similar datasets and operations, or by setting these values in a trial-and-error approach till the best performing solution is found (in terms of higher model accuracy). This trial-and-error approach is part of the hyperparameters tuning phase of the model construction, in

which the number of hidden layers and hidden neurons represent, indeed, two of the hyperparameters of the network to be set.

3. Output layer.
   Last layer of the network, which conveys the information learnt by the network in the hidden layer(s). The output values are related to the prediction or classification on the input data and can be continuous, binary, or categorical variables.
   In case the network is used to perform a non-binary classification, the output layer contains as many neurons as the number of possible classification labels, that is, one for each possible class that the non-binary dependent variable can assume. In case of binary dependent variable, instead, one output neuron is enough, since it can assume one value or the other depending on the predicted class for that input.

So, when data enters the network as input, this input will undergo processing by each single node within the network: as input data is passed to a layer, the nodes in that layer process such data and then pass the obtained output to the following layer as input.

The operation taking place within a specific node depends on the kind of layer it belongs to.

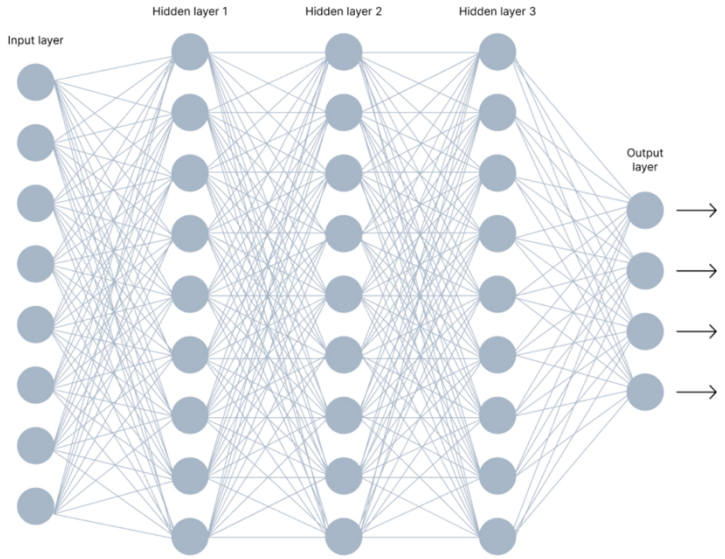

Figure 3.29 - Example of deep ANN with 3 hidden layers [170]

Nodes are connected through weighted links (synapses) and the signal about data processed by nodes in a certain layer is transferred to nodes in the next layer through these connections, until the output layer is reached. The weight on the link connecting two nodes can be seen as the strength and importance of such connection.
As the data moves from the input layer to the first hidden layer, or from a hidden layer to another one, all values entering a neuron in the next layer are added up into a weighted sum of the input values, where weights are the synapses' weights. Then, a transformation function assigned to the specific neuron or to the whole layer (called activation function) is applied to the weighted sum and the result of this transformation is passed on to the next neuron.

This procedure is repeated throughout the whole network many times, depending on the network dimension (number of neurons, layers, and synapses), until the final output layer is reached.

Besides the distinction between input, hidden, and output layers, there are different typologies of layers which can be outlined, depending on their characteristics and on the kind of transformation they apply on the inputs. In the following, 3 examples are reported [171] [172]:

a. Dense layer or fully connected layer. Each node belonging to such layer is connected to every neuron in the previous layer and every neuron in the next layer, meaning that each neuron in the layer receives, as inputs, the outputs from each of the neurons in the preceding layer, and sends its own output as input to each of the neurons in the following layer. If the network is composed of fully connected layers only is called fully connected network.
   Typically, hidden layers in an ANN are fully connected.
b. Convolutional layer. It is the main element composing a Convolutional Neural Network, which processes datasets of images. It contains a number of filters to be applied on images during the convolution operation.
c. Pooling layer. It performs the Pooling operation typical of a Convolutional Neural Network.

3.3.2 ANN deployment steps: training, validation, and testing

For the ANN to be deployed, 3 main steps need to be followed. They are listed in the following and described in the next paragraphs:

1. Training process to iteratively optimize the predictions by converging towards the optimal weights on the synapses linking the neurons.
2. Validation and hyperparameters tuning, for choosing the best model configuration.
3. Testing over new, unseen, and unlabeled data to finally evaluate the performance of the algorithm to real applications.

Synapses (links between neurons) are assigned weights, that are at the basis of the ANN learning process, which consists in solving an optimization problem related to the optimization of the weights on the network's connections. The model, indeed, learns the optimal values to assign to the weights based on how changes in their values affect the error in the model predictions.
This occurs by passing the training dataset to the network several times iteratively, so that it can learn from it, and the main training steps are listed in the following:

1. The synapses weights are randomly initialized to values close to zero (but not zero). Eventually, some weights on synapses can be assigned a zero value, in case the value entering as input in a node, coming from a previous unit, is not relevant for the specific prediction.

2. The input layers are fed with the features (one per input node) of one single observation in the training dataset (e.g., one single image or one single row in a table).

3. Forward-propagation process.
   Input values entering one node (in the first hidden layer) are added up into a weighted sum (where weights are those of the synapses entering the node) and this partial result is called pre-activation output of the node.

$$Pre - activation\ output = \sum_{i=1}^{m} w_i x_i$$

m: number of synapses entering the node
x_i: an input value entering the node
w_i: the weight on the link through which x_i is entering the node

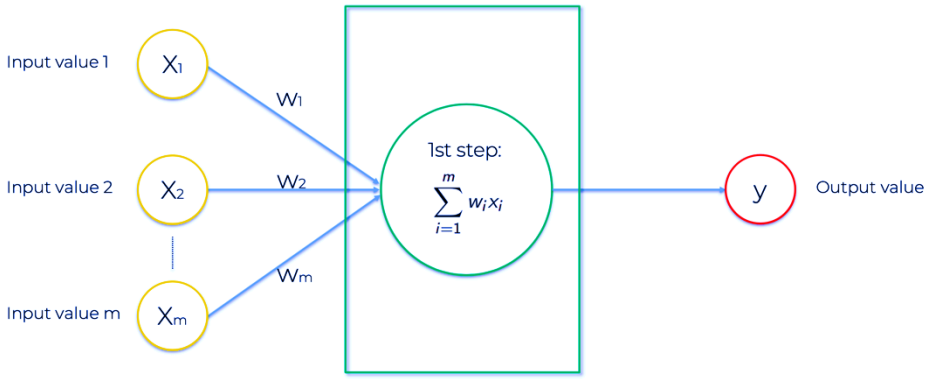

Figure 3.30 - Computation of the pre-activation output by a hidden neuron [121]

This weighted sum is processed by the node by applying the so-called activation function which gives a value corresponding to the final activation output of the node.

$$Activation\ output = f\left(\sum_{i=1}^{m} w_i x_i\right)$$

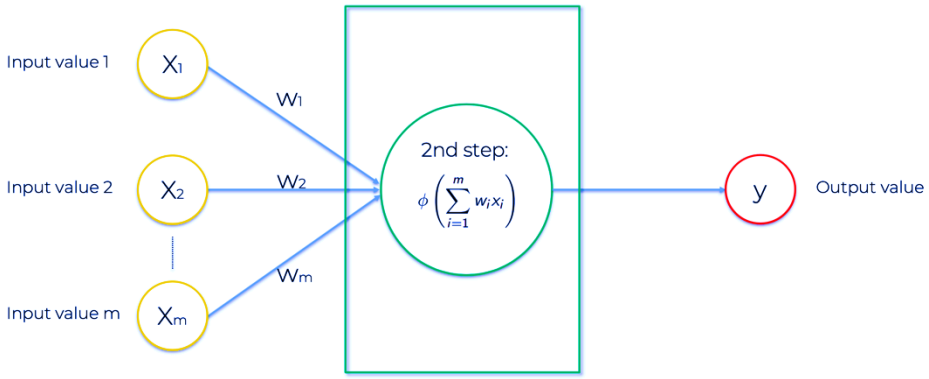

Figure 3.31 - Application of the activation function by a hidden neuron [121]

Based on this final result, the neuron understands whether the current signal is worth being passed on to the next neuron or not for the specific input received. In other words, the activation output defines whether the neuron itself is to be activated and the output sent to the next neurons down the line, or not activated and no data passed

along (activation output = 0). Indeed, not all nodes are important to identify the prediction of a specific input observation.
This process takes place within all hidden neurons.

So, the information is entered into the input layer and then it is propagated forward through the network until the output layer is reached and the prediction output is generated for the given input. Output layers, as well, apply their own application function to the input they receive and, since each output neuron corresponds to a specific class that the dependent variable can assume, the output node with the highest activation value is the one giving the class to which the input is predicted to belong.

4. The predicted results (i.e., what the input is predicted to be) are compared with the 'ground truth', that is, the actual target values of the dependent variable (i.e., what the input class actually is) found in the labeled dataset, and the generated error is computed through a cost (or loss) function, which indicates the error in the prediction and, as such, how well the model performs predictions on the training data. The goal of the ANN training phase is that of minimizing the value of this loss function as much as possible so that the value of predicted variables is as close as possible to the actual values.

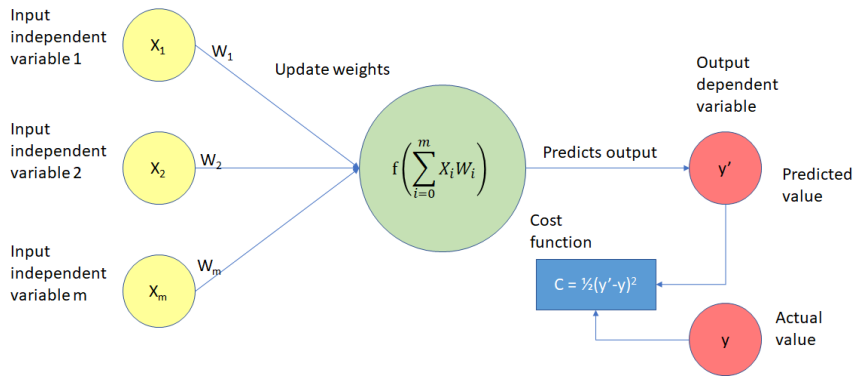

Figure 3.32 - Comparison with the 'ground truth' and computation of the cost function [173]

5. Backward-propagation process.
The error information is backpropagated through the network, so that the network can be optimized by adjusting the weights in a way to minimize the cost function.
Weights are adjusted based on advanced optimization algorithms, among which the most widely used are the stochastic gradient descent method (SDG) or the batched gradient descent method (BDG).
For a specific individual data sample, by looking at the resulting output values from the output nodes, the objective of SDG and backward propagation is that of increasing the activated output value generated by the correct output node (to which class the specific input data actually corresponds to) and decrease that generated by all the other output nodes, so to overall decrease the loss function. Intuitively, this can be obtained by going backward through the network layers, since the output of each layer in the network depends on the weights and output values of the previous layers: the output values, indeed, result from weighted sums computed in the previous layers

of neurons, and in order to indirectly modify the output nodes values in the desired direction, the weights on synapses should be updated (influencing all following layers till the output one).
The final result consists in obtaining the optimal values for these weights which most accurately map the inputs to the correct output prediction or class.

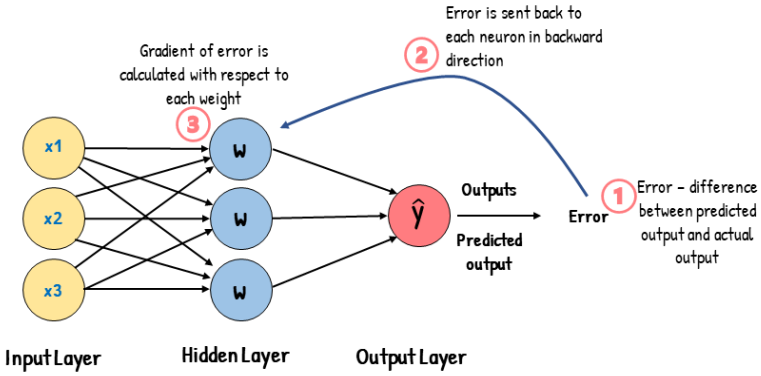

Figure 3.33 - Back-propagation and SGD [174]

6. So far, only one datapoint (or observation) has been passed to the network as an input. Now, all the steps are repeated, and weights are iteratively updated, passing to the network all the other observations in the dataset.
   An epoch is defined as one single passage of the whole training set through the ANN, and typically, the model is trained through multiple epochs for the neural network to keep on adjusting itself and allow for the accuracy of predictions to improve over the epochs.
   Weights can be updated after the passage of each single observation through the network, or after the passage of the whole training set (i.e., an epoch), depending on the method used (stochastic gradient descent or batched gradient descent respectively).
   In this way the cost function is minimized: weights are optimized as the model keeps on learning from the data, and iteration by iteration the model gets more accurate.

ACTIVATION FUNCTION

The activation function of a neuron plays a crucial role in the forward propagation process, that is the one of defining the output of the neuron to be passed over to the next node, based on the input received. The weighted sum of each input into the neuron (defined as pre-activation output of the neuron) is passed to an activation function, which applies a non-linear transformation to the data and the resulting activated output will be passed as input to the next layer.
In other words, the activation function allows to establish whether a neuron's output is relevant for the network to perform the specific prediction or not and so, whether the node itself should be activated or not (in the same way as specific groups of biological neurons are activated following a certain stimulus). If this is the case, the node is ignited, and the activation function defines the output of that node starting from the set of input values (outputs of the previous layer). For example, if the input sample is an 'apple', only some nodes will be ignited in the network, that are those nodes needed for identifying the input as an 'apple'; the same

in the case the input is an ‘orange’: only some nodes will be activated to predict that the corresponding output is ‘orange’.

There exist different typologies of activation functions, and here the 5 mainly used ones are described:

1. Threshold (or binary step) function.
   The output can be 1 or 0, and so the neuron is activated or not based on a threshold value for the input fed to the activation function (i.e., the pre-activated output of the node). If the input value is higher than the threshold level, the neuron is activated and the output of the function is passed on to the neuron(s) in the following layer, otherwise the neuron is disabled (the output is not transmitted to the next layer, i.e., the output is equal to zero).

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geq 0 \end{cases}$$

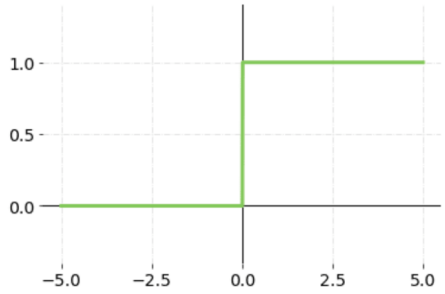

Figure 3.34 - Threshold activation function [175]

2. Sigmoid (or logistic) function.

   The input values are transformed into output values included within the 0-1 range. Such function corresponds to an s-shaped curve, asymptotically approaching 0 the lower the input value, and 1 the larger the input value. So, the more negative the pre-activation value, the more the sigmoid function will transform it into a number close to zero (lower limit); while the more positive the input value, the more the sigmoid function will transform it into a number close to one (upper limit).

   The closer to 1 the output value is, the more the specific node is activated, while the closer to 0 the output value is, the less the specific node is activated: so, some neurons are more or less ‘fired’ than others.

   Sigmoid activation function provides the probability of activation for the considered neuron. In the classification models realm, this means that it can provide the probability of an input data to belong to one class or to another. For this reason, this activation function is typically used in the output layer for binary classification, and it returns the predicted probability of the variable to belong to one of the two classes, that is, the probability of the single output neuron to be activated or not.

$$f(x) = \frac{1}{1 + e^{-x}}$$

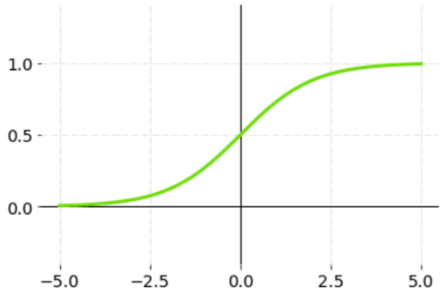

Figure 3.35 - Sigmoid activation function [175]

3. Hyperbolic tangent (tanh) function.

   Similar to the sigmoid activation function, but in this case output values can be below zero as well, ranging from -1 to 1.

$$f(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$$

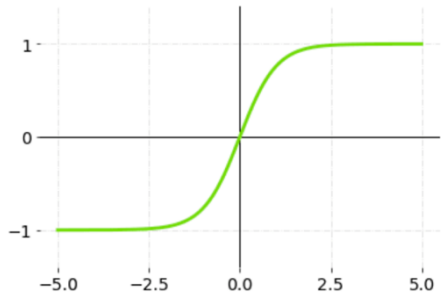

Figure 3.36 - Hyperbolic tangent activation function [175]

4. SoftMax (or Normalized exponential) function.

   This function is a generalization of the sigmoid function and is mainly used in the output layer of the network in multi-class classification problems, since it returns the probability of the data sample to belong to each possible category i.

$$f_i(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \; for \; i = 1 \ldots . K$$

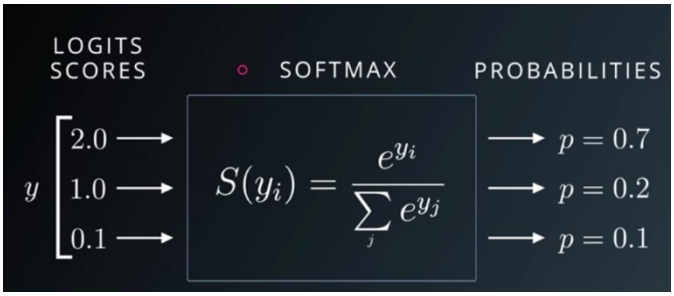

Figure 3.37 - Softmax activation function [176]

Sigmoid function is preferred to Softmax function in case of binary classification problems where the output layer is composed of one individual neuron. Indeed, both functions take as input the raw results by the neural network (logits) and give it a practical probability meaning by normalizing them (i.e., by mapping them into the probability of that datapoint to belong to a certain class). However, Sigmoid outputs a single probability of the data sample to belong to Class A (so to implicitly derive the complementary probability to belong to Class B); while SoftMax receives as inputs a vector containing the raw results by the network and outputs a vector containing the probabilities of the data sample to belong to each possible class. It basically normalizes the otherwise raw outputs by the network into a probability distribution, in a way that they all are positive, lower than 1, and sum up to 1 (so to be interpreted as probabilities of belonging to the different classes) [176] [177].

5. Rectifier function or ReLu (rectified linear units) function

   It is the most popularly used activation function in deep learning algorithms.
   This function transforms the input to the maximum value between zero and the input value itself: so, it maps any negative or zero input value into a zero output, while any positive input value will be mapped to the value itself.
   The more positive the input value, and so the output value, the more the specific neuron is activated.

$$f(x) = \max(0, x)$$

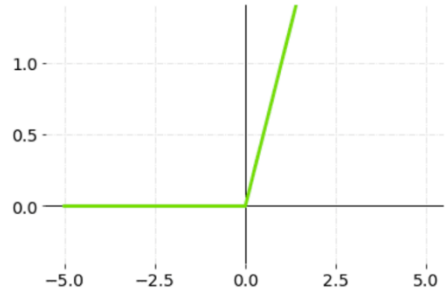

Figure 3.38 - Rectifier activation function [175]

Typically, in an artificial neural network the ReLu activation function is used on the hidden layers' nodes, while the sigmoid or SoftMax activation functions are applied on signals incoming to the output layer neurons in order to be able to predict a probability as final output of the network.

What is crucial is for the activation function to be non-linear. Indeed, a composition of linear functions is linear itself: since the pre-activation output of a node is obtained through a weighted sum, that is a linear transformation of the data, if also the activation functions were to be linear then the overall transformation on the input data would be linear. In such case, the mapping learned by the network from input to output would be linear as well, even if the network was very deep (i.e., composed of a large number of hidden layers). However, networks are usually needed to learn complex non-linear functions and perform non-linear

mappings, that is, to model a target variable which varies non-linearly with its independent variables, and this is made possible by the use of non-linear activation functions.

STOCHASTIC GRADIENT DESCENT (SDG) METHOD AND BACKPROPAGATION

In order to find the optimal weights for the synapses, the neural network follows an optimization algorithm: the batched gradient descent (BGD) method or the stochastic gradient descent (SGD) method [121] [178] [179]. The overall aim is that of finding the model parameters (weights) corresponding to the best fit between predicted and actual output values. In other words, the objective is to minimize the cost (or loss) function computed at the end of the forward propagation process, by iteratively updating the synapses weights within the network in its training process, so to make the model as accurate as possible. The cost or loss function is defined as the error between the prediction carried out by the network on the training dataset and the ground truth about that prediction (i.e., the actual value), and it can be minimized by changing the model weights. This function can assume different forms: for instance, SSR or MSE (i.e., sum of squared residuals or mean squared error) can be used in case of a regression model, cross-entropy loss function can be used for a classification problem where output variables are binary, and so on.

The value of the loss function is calculated after each epoch or after each single observation in the training set passed through the network, depending on whether the method applied is batched gradient descent method (BGD) or the stochastic gradient descent method (SDG) respectively. In the first method, the whole training dataset is passed to the network at once, and the error made by the network in predicting each datapoint in the training set is summed up so that the model is updated through this overall result only after an epoch is concluded (all training samples have been predicted). It is efficient in terms of computation but shows long processing time and is usually applied just in case the cost function is convex, since otherwise it might end up finding a local minimum rather than the global one.
While the second method takes a single data sample, feeds it into the network, computes the cost function and updates the weights accordingly: it can be generally applied to all kinds of cost functions and makes it much more likely to find the global minimum. For this reason, it was chosen for the case study presented in the final chapter of this thesis work.
There is also a middle-ground method called mini-batch gradient descent method: a batch containing a certain number of training samples (lower than the whole dataset and called mini-batch) is used to compute the cost function and update the weights at each iteration.

Focusing on SDG [180], after the cost function C is calculated, the method implies the computation of the gradient of the loss function (i.e., vector containing the partial derivatives of the function) with respect to each single weight that has been assigned to the network links, going backward through the network layers with the backward propagation approach.

$$\boldsymbol{w} = (w_1, w_2, \dots, w_n)$$

$$\nabla C = (\partial C/\partial w_1, \partial C/\partial w_2, \dots, \partial C/\partial w_n)$$

The meaning of this gradient is to see how much the loss function changes when one single weight in the network changes, i.e., how much the loss function is affected by the change in a specific weight.

Starting from a vector of arbitrarily chosen weights $\boldsymbol{w}$, the objective is to update it so to move iteratively towards the direction of fastest decrease of the cost function, that is the direction of the negative gradient $-\nabla C$ (i.e., what is the weights update that would cause the largest decrease in the cost function?).
To do so, the individual weights are updated by subtracting, from the previously assigned values, the value of such gradients multiplied by the learning rate, that is a very small number (usually included within the [0.0001-0.01] range). The learning rate determines how large the update is and so, how big the iteration step is.

$$\boldsymbol{w} \rightarrow \boldsymbol{w} - (\nabla C \times learning\ rate)$$

So, weights are updated in a way that the cost function is reduced in the most efficient way possible: indeed, each weight is updated based on which portion of the error it is responsible for (i.e., taking the gradient). A certain weight is updated by a specific quantity, relatively to the other weights, that is higher or lower based on the magnitude of the effect that such update will have on the loss reduction.

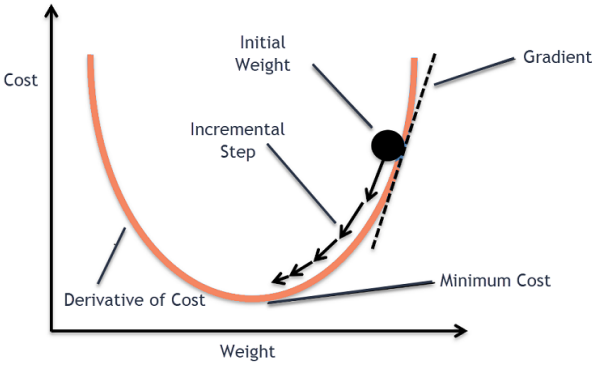

Figure 3.39 - Gradient Descent method [181]

The same training dataset is iteratively passed to the neural network through several epochs, so that the model can be trained, learn, and update the cost function, as well as the weights on the links, over several iteration, so that they incrementally reach their optimal loss-minimizing values.

The convergence of weights to their optimal values and loss function to its minimum occurs over several steps, whose size is dependent on the set learning rate: the learning rate defines by how much the weights are updated at each iteration, and it is a hyperparameter which needs to be tested in order to be tuned and find its best value. The higher the set learning rate, the higher the amount subtracted to the old weights and so, the higher the risk to overshoot the minimum of the cost function, while the lower the learning rate, the more the steps in the search process for the cost function minimum and the lower the chance to miss it. However, in this second case, convergence might be too slow and never reached within the training over the fixed number of epochs.

To have a clearer understanding of the whole training process steps, a general flow chart (*Figure 3.40*) is represented and then further decomposed into a specific flow chart for the forward and backward propagation processes (*Figure 3.41*).

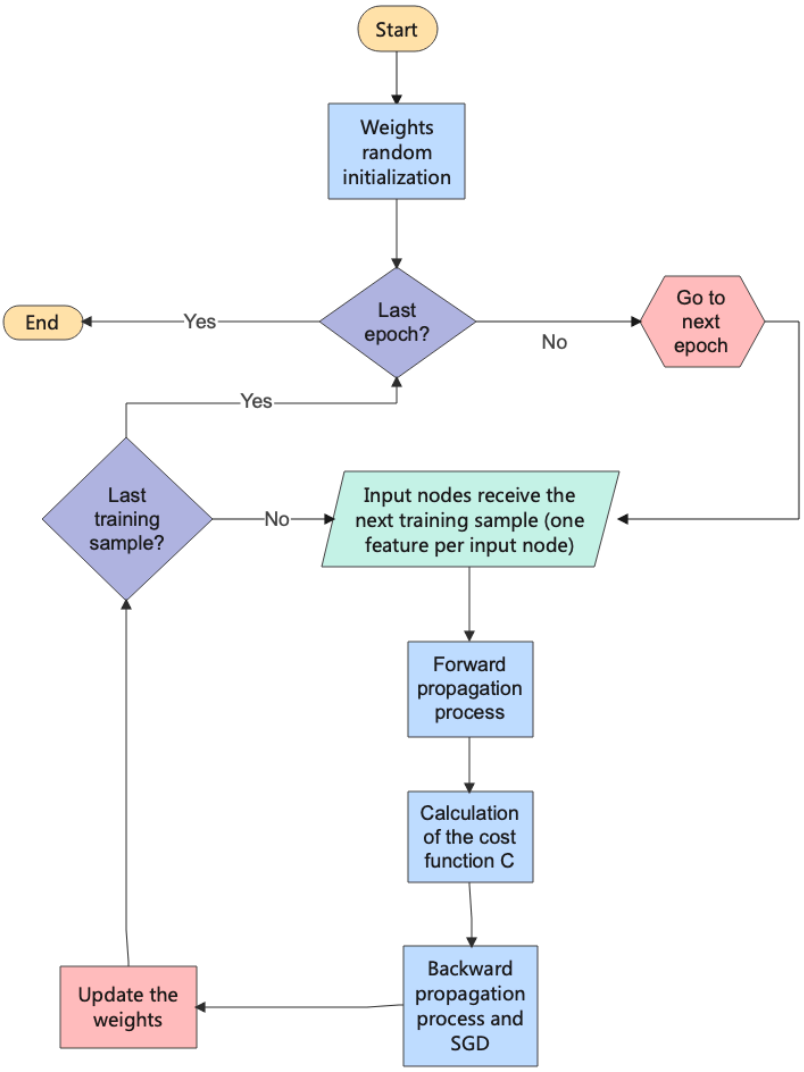

Figure 3.40 – ANN training process (with SDG) flow chart

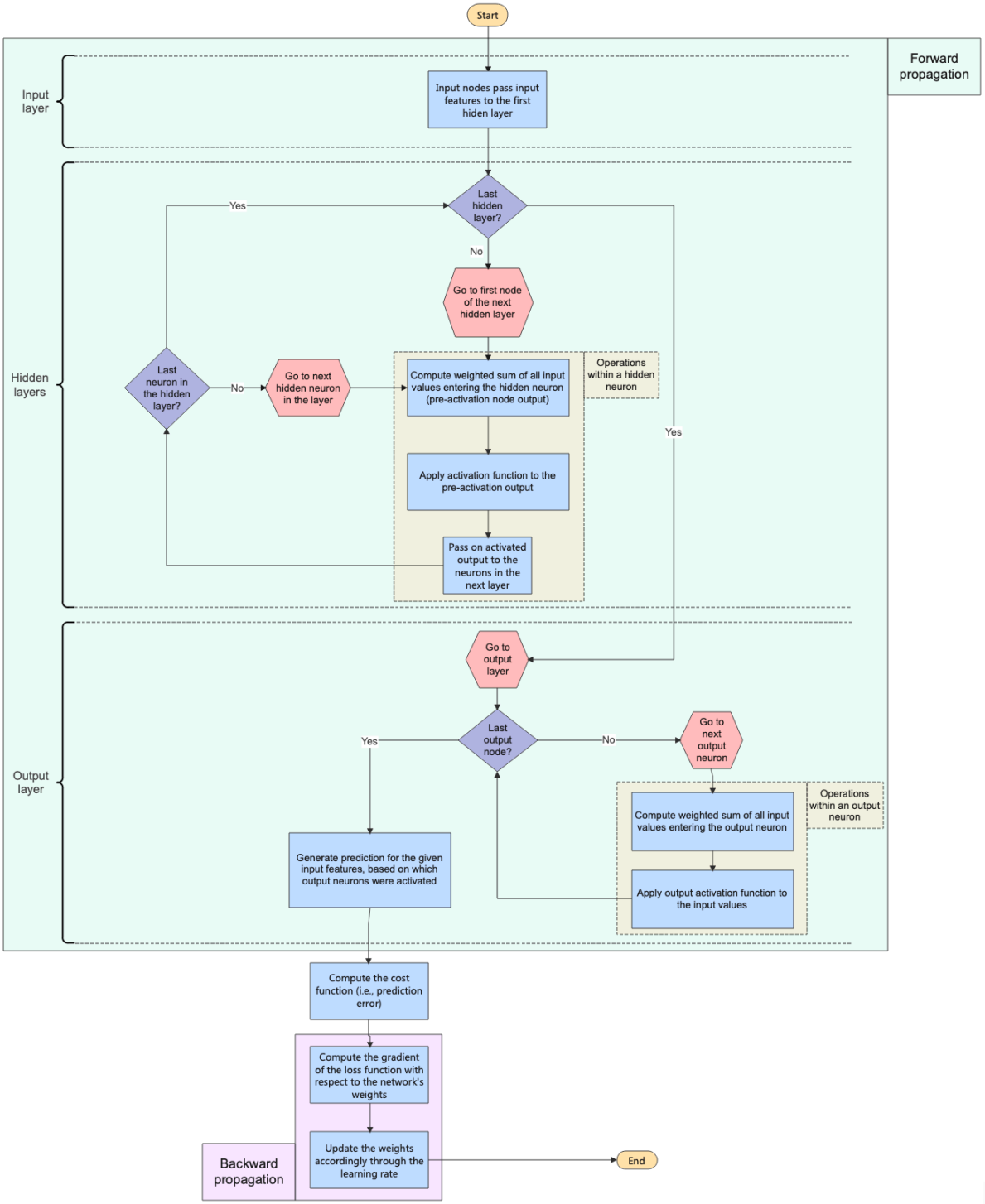

Figure 3.41 - Forward and backward process broken down, flow-chart

VALIDATION AND HYPERPARAMETERS TUNING

The most suitable model configuration needs to be chosen, in terms of model's prediction accuracy. The performance of a specific model configuration used for the training process can be evaluated by making the model run predictions over the validation set. As mentioned in other portions of this document, hyperparameters tuning allows to actually 'tune' the values assigned to the hyperparameters of the network, by running several times the training process with different model settings and see the performance of such configuration by

making the model compute predictions over the validation set. On the basis of the validation accuracy obtained by different model configurations, the most appropriate one is selected. Validation can be performed by simple 'hold-out' methodology (i.e., by simply selecting a subset of the initial training set to play as validation set) or through more sophisticated methodologies, like K-fold cross-validation, which will be detailed in the practical application developed within this thesis work.
The tuning task can be performed by manually attempting different model configurations, training the model for each of them, and calculate the model accuracy over the validation set for each single attempt; or automatically, thanks to specific methods included within some libraries which perform different model configuration attempts by iteratively changing the values assigned to hyperparameters.

TESTING

When the best model configuration is finally obtained through hyperparameters tuning, then, the model is ready to be deployed in real-life cases. To have a final verification of its accuracy, it is possible to make the network perform predictions over the test set, which is unlabeled and, as such, simulates a real case scenario, in which the network will be fed with real unlabeled input data, of which the classification or prediction is needed to solve the specific problem of interest.

3.4 Convolutional Neural Networks (CNN)

3.4.1 Computer vision

Computer vision [182] [122] [121] is a subfield of artificial intelligence which refers to the ability of machines to understand, analyze, and process visual data (e.g., images and videos) more efficiently than human beings can do. Through computer vision, machines gain the ability to see and observe visual data, being capable of collecting meaningful information from it. This is possible thanks to large amounts of data and the use of deep learning algorithms and neural networks (in particular, Convolutional Neural Networks).

Computer vision can perform many different tasks, and below the 4 most diffused today are listed:

a. Image recognition or classification. The network receives images as input and is able to classify the images into pre-defined categories.
b. Object detection. The network can identify pre-determined elements within the images.
c. Image segmentation. The network can partition images into pre-defined segments.
d. Image generation. The network generates images based on some pre-defined categories.

3.4.2 What is a Convolutional Neural Network?

A Convolutional Neural Network [122] [183] is a specific typology of Artificial Neural Network mostly used to perform images analysis and classification and carry out other computer vision tasks. In the same way as the human brain processes visual information by looking for visual features, a CNN is able to identify patterns and to process visual features, so to be able to categorize images much faster than with the traditional manual and time-consuming feature extraction methods.
What characterizes the CNN with respect to a regular ANN are the convolutional layers, which receive image inputs, transform them through the so-called convolution operation, and pass them onto the next layers till reaching the output layer, in which the image class label is derived. The convolution operation maps a given input into an output and is what allows the CNN to detect patterns in images through a selected number of filters for each convolutional layer.
For instance, a pattern to be detected could be edges in the image and so, the specific filter would be called edge detector; or circle detectors, corners detectors, etc. The deeper the convolutional layers, the more sophisticated elements and objects they are able to detect.

A Convolutional Neural Network works through 4 major steps:

1. Convolution.

   As anticipated, this operation takes place within the convolutional hidden layers through the help of a certain number of filters, also called feature detectors or kernels, which are able to detect patterns in the input images.
   Keeping in mind that the input image is saved by the machine as a matrix containing data related to individual pixels, the filter is a $n \times n$ matrix (whose size can be chosen) that is applied on the input image and covers step-by-step (the step size is called 'stride', and it is measured in pixels) all portions of such image, moving over it until it covered it entirely.
   More analytically, the filter covers the first $n \times n$ pixels portion of the image and an element-by-element product is computed between the filter matrix and the $n \times n$ matrix reporting the pixel related to that covered portion. Then, the filter slides over each $n \times n$ set of pixels in the image (this sliding process is called 'convolving') moving by the set stride: the result of each matrix product is stored in a new matrix called feature map (or convolved map or activation map).

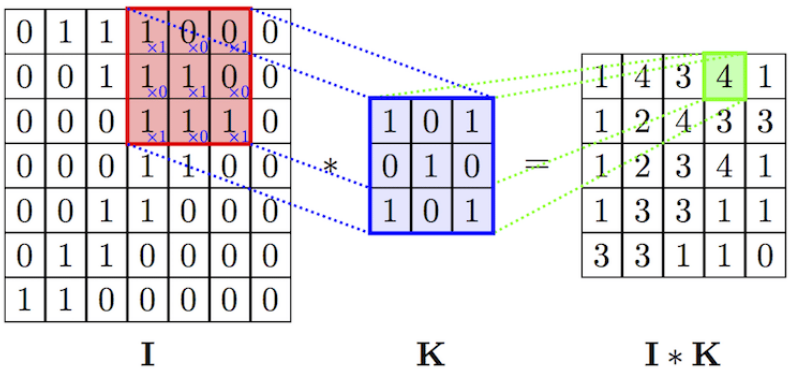

Figure 3.42 - Simple example of convolution operation (I = input image, K = 3x3 kernel) [184]

In the figures below, some convolution steps are displayed as an example. A 3×3 left-edge filter is applied and the stride is fixed to 1: on the left it is possible to see the original image and the filter sliding over portions of it, while on the right the feature map resulting from the matrix products.

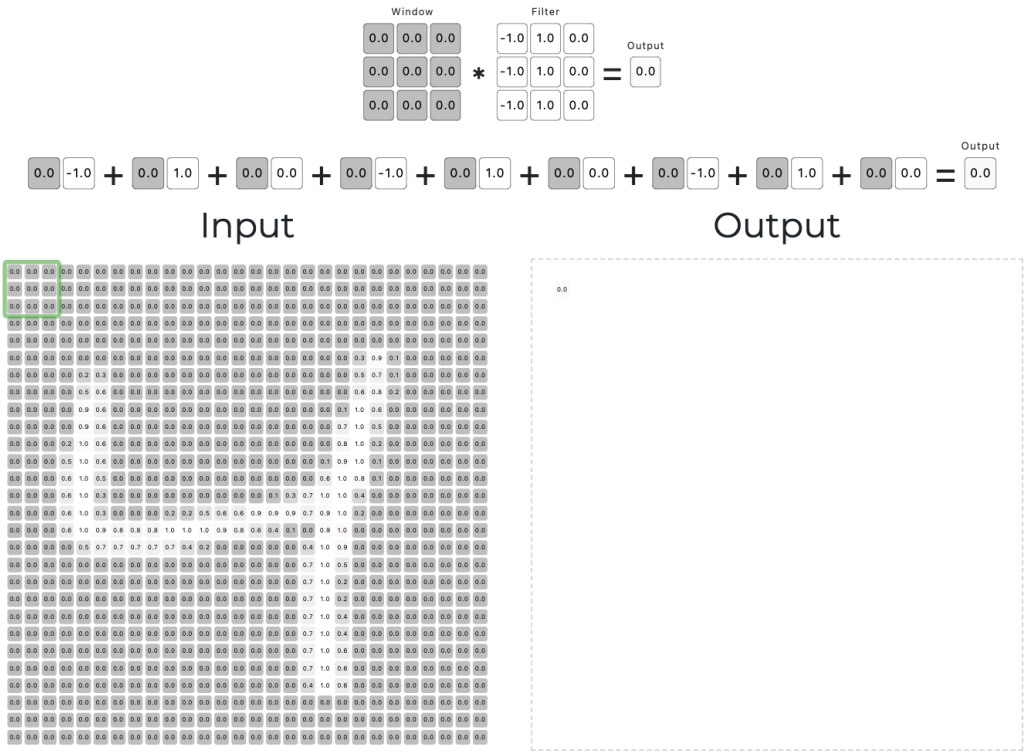

Figure 3.43 - First convolution step [185]

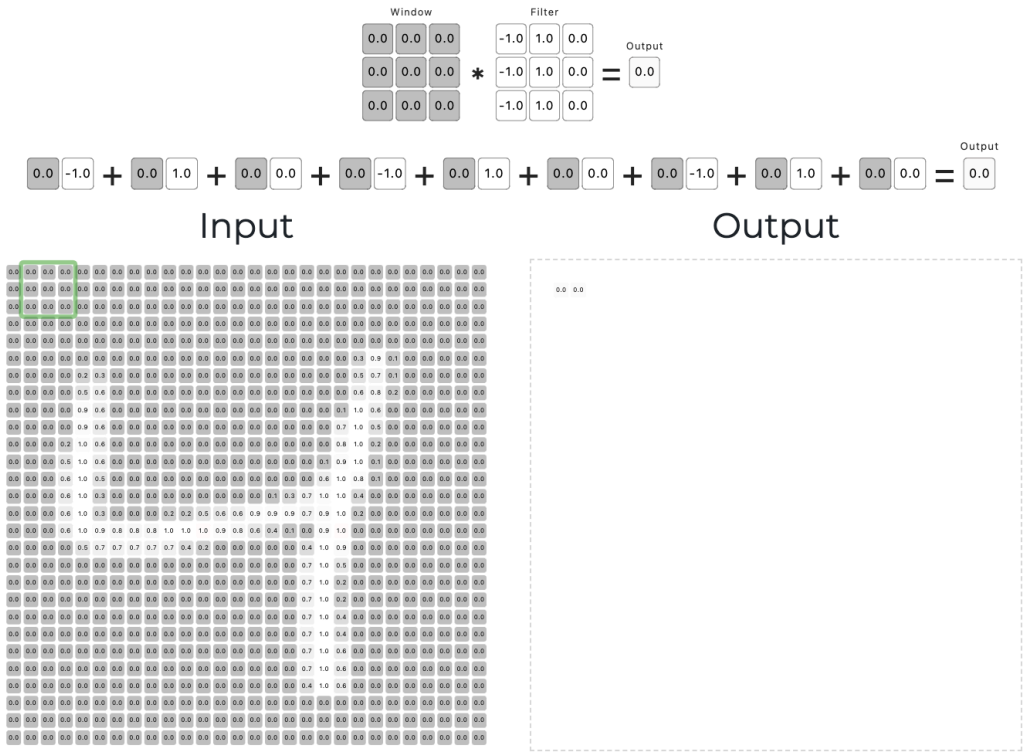

Figure 3.44 - Second convolution step [185]

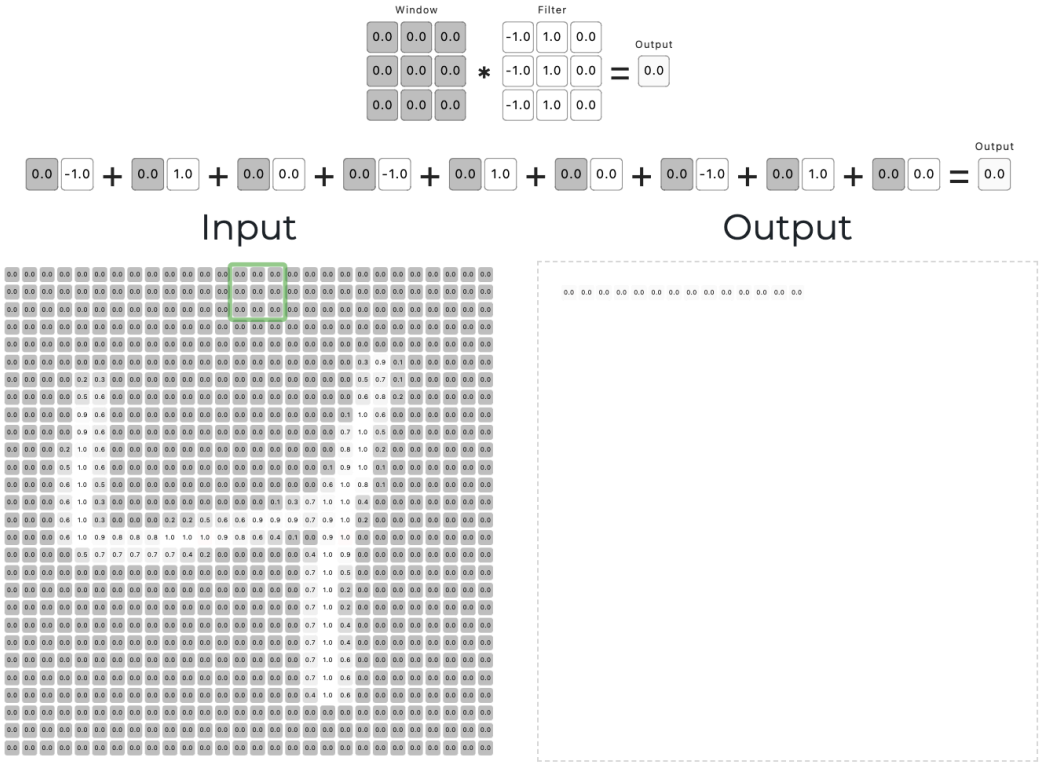

Figure 3.45 - Fourteenth convolution step [185]

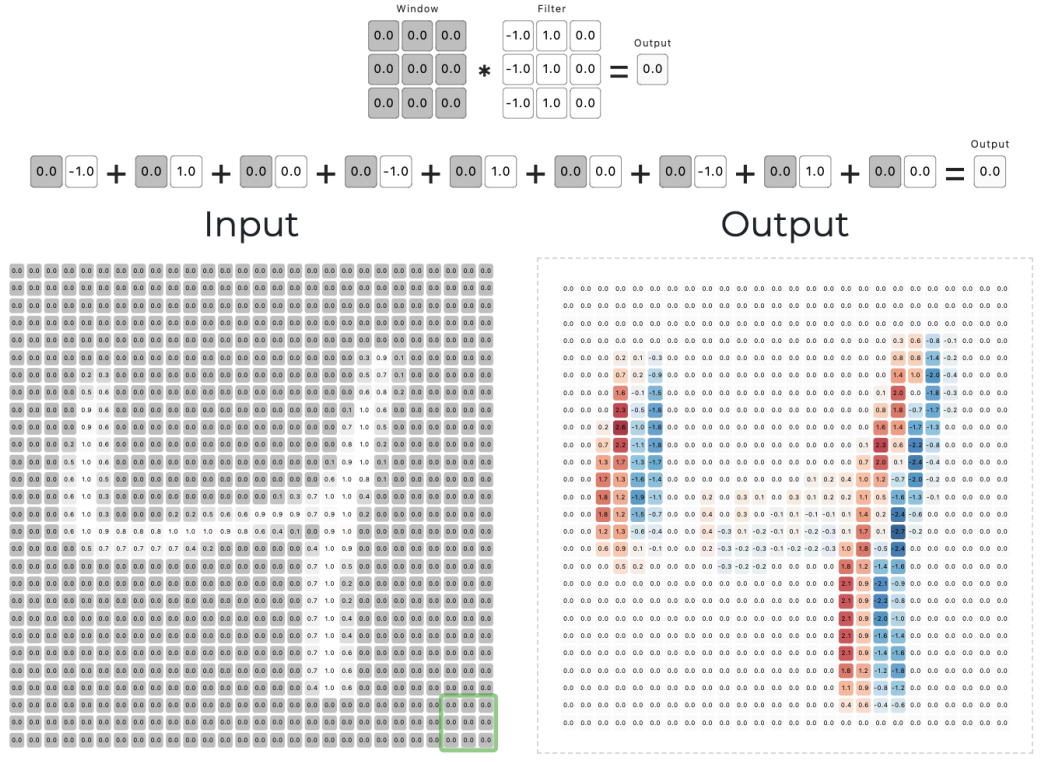

Figure 3.46 - Final feature map [185]

So, as soon as all blocks of pixels are convolved by the filter, the result is this new matrix representing the input image, which is of smaller size than the initial image, since the application of the feature detector causes some information loss. The purpose of the feature detector, indeed, is to detect certain features on the image and

generate a feature map that summarizes the presence of the detected features: the filter is characterized by a specific pattern, and the highest numbers appearing in the feature map corresponds to portions of the image in which the specific filter pattern was most highly matched up by the input image. This convolution process, indeed, extracts and preserves specific features from the image, by getting rid of what is not necessary for the analysis.
Supposing the image size to be given by $f \times f$ pixels, and the filter size to be $n \times n$, then the size of the resulting feature map will be given by:

$$filter\ map\ size = (f - n + 1) \times (f - n + 1)$$

Yet, if the image goes through many convolutional layers and gets convolved by a certain number of filters, the output image keeps on getting smaller and smaller and such information loss can be dangerous, causing the images to become meaningless by losing valuable data.
To face this issue, Zero Padding technique [186] allows to preserve the original input size even if convolution operation is performed. The technique implies the addition of zero-valued pixels frame around the input image edges: it is said that the input images is 'padded' with a border of pixels with value equal to zero.

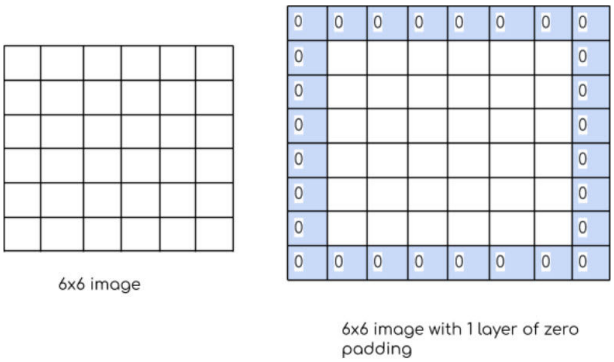

Figure 3.47 - Zero padding method [187]

Depending on the size of the specific input image and filter, it might be necessary to add more than one zero-pixel border around the image in order to be able to maintain the initial input image size.

It is the network itself during its training process which decides what features are important to be extracted, and then looks for these patterns through the application of different filters (one per feature), obtaining different feature maps on the image, one per specific feature. The number of desired filters per convolutional layer has to be specified (since it is a hyperparameter), fixing in this way the number of resulting feature maps generated per convolution layer.
Each feature map containing the result of the convolutions on the input image is the output of the specific convolutional hidden layer and will be passed as input to the next layer: in the following convolutional hidden layers, if any, this input matrix will undergo the same filtering process.

After each single convolution layer, a ReLu activation function is applied to add non-linearity to the CNN, in line with the fact that images are highly non-linear (i.e., containing varied pixel values): the transition between adjacent pixels in an image is non-linear, since each of them is characterized by different elements, colors, borders, etc.
A convolutional layer by itself would be linear (as well as a fully connected layer is) since its output is simply an image with a filter passed over it (i.e., a linear operation), and with the application of the ReLu function is possible to apply non-linearity to the feature maps: the operation is applied to each pixel of the feature maps and it replaces the negative pixels values with a zero value, so to 'eliminate' negative values by setting them to zero while isolating the important features.

2. Max pooling.

Max pooling operation takes place after each individual convolutional layer and is a transformation applied on the feature maps. A $n \times n$ filter is chosen and placed on the first $n \times n$ region of the feature map, which can be seen as a pool of $n \times n$ numbers. The maximum value included within this $n \times n$ box is recorded and the filter slides over the rest of the convolution output according to a selected stride (i.e., sliding step in pixels units), repeating the same process. All the maximum values extracted from each pool are recorded into the so-called pooled feature map, that is a new representation of the image and the output of the max pooling operation.

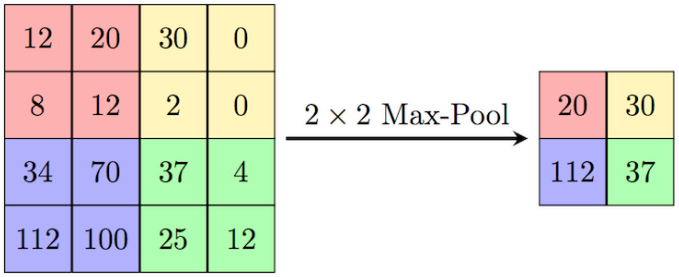

Figure 3.48 – From feature map to Pooled feature map (3x3 filter, stride = 2) [188]

Max pooling operation is beneficial from different perspectives:

a. Through max pooling the highest numbers are selected and since the higher valued pixels on the feature map are the ones in which the closest similarity to a feature was found by convolution, the features of interest are preserved going forward to the pooled feature map. This is exactly the network purpose: trying to extract some specific features from images (e.g., curves, edges, etc.) and being able to detect them even if distorted (e.g., rotated).
b. It reduces the dimensionality of the image by reducing the number of pixels with respect to the output of the previous convolutional layer. Consequently, the complexity reduces: indeed, the number of parameters in the CNN is lower as well as the computational load, by keeping only valuable information.
c. As a consequence of previous point, reducing the number of parameters allows higher overfitting prevention, since the 'noise' (not relevant information) is set aside.

There are different types of pooling, like min pooling, sum pooling, average or mean pooling, etc., but max pooling is the most used in practice.

3. Flattening.

Having the pooling layer with many pooled feature maps, the next step consists in flattening all these maps into one single column sequentially.

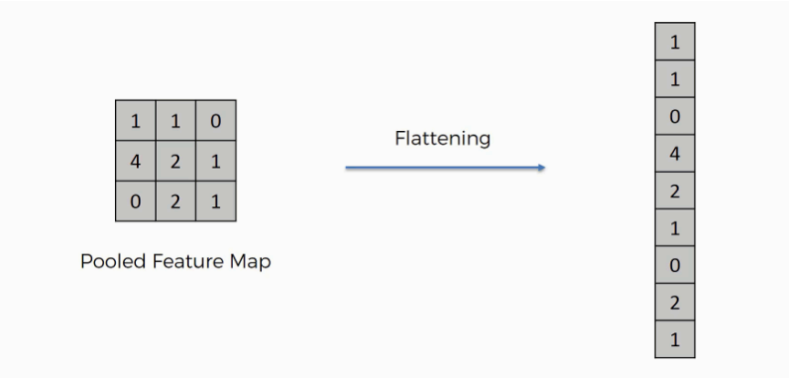

Figure 3.49 - Flattening operation [189]

The reason is that this vertical feature vector is the format that will be required in the following step as input to feed an ANN for further processing steps over the features.

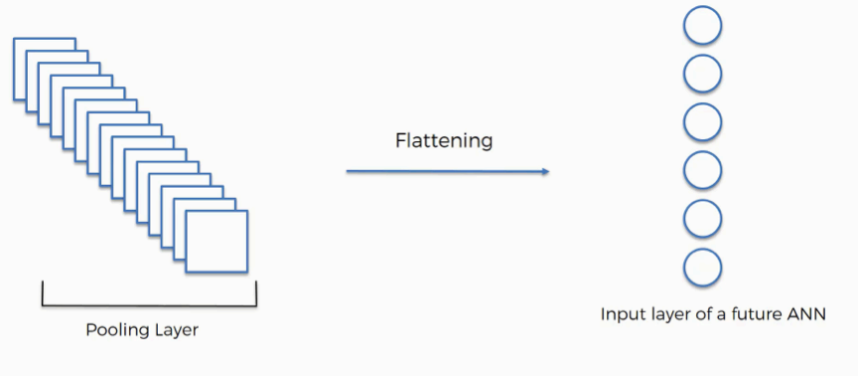

Figure 3.50 - Feeding the ANN input layers with the flattened features [189]

4. Full connection.

Finally, a whole ANN is added to the layers built so far (convolutional, pooling, flattening) with its input layer (fed with the outputs obtained from the flattening procedure), fully connected hidden layer(s), and output layer (one per image classification category).
Indeed, besides image processing through convolution, pooling, and flattening, the network undergoes the same training process as a classical ANN. Supposing an image classification problem (like the one in the case study presented in the last chapter), a prediction about the image class is made and an error is calculated through a cross-entropy function, describing how good the network is performing. In the attempt of minimizing the loss function, the error is backpropagated through the network and weights are adjusted to optimize the performance. In case of a CNN, though, also

feature detectors are adjusted: indeed, bad performance could also be due to searching for the wrong features.
Over the training iterations, the output neurons (1 or more, depending on the number of existing image classes) understand which weights to assign to all the synapses connected to them in a backward propagation fashion, so that they learn which of the previous fully connected layer neurons convey important information for the specific image category they are assigned to predict. Each neuron in the previous layer is associated to a set of features, and it is up to the output layer to decide whether the signal emitted from a previous neuron, and so its associated features, is related to the image class it is looking for.
The network predicts probabilities of the image to belong to each category, and its first choice is the class corresponding to the output neuron yielding the highest predicted probability.

4. The welding process

Welding is a manufacturing process which consists in joining two or more components into larger assemblies by fusing them together through heat, pressure, or both, generating a strong and permanent bond once the materials cool down. The procedure is usually performed on metals and thermoplastics [190] [191] [192].
Various types of welding processes can be distinguished, on the basis of the process technical features and areas of employment. Throughout this thesis work, the welding processes among the most diffused ones are going to be mentioned, but to make the following chapters more readable, they are first described in this section.

4.1 Resistance Spot Welding (RSW)

Resistance spot welding process [193] [194] [195] is a fusion welding process that is used to join two or more metal sheets (usually thin) by overlapping them and producing a spot weld in that overlapped surface, through the application of pressure and heat to the weld area by 2 electrodes.
Pressure is applied between the two electrodes and the components to be welded (to contain the workpieces), a large electrical current is conveyed from the electrodes to the piece, and the resistance heat that is generated (due to the electrical current combined with pressure) melts the sheets forming a molten nugget and consequently the weld spot [81]. This area that is generated in the point in which the two metal sheets are joined is called weld nugget.

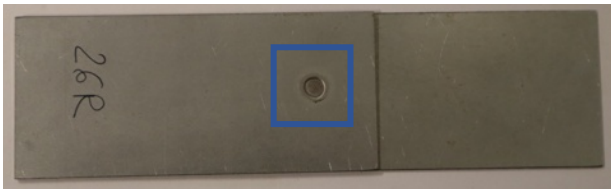

Figure 4.1 - Example of nugget

The procedure works based on Joule's law of heating, according to which heat is generated proportionally to the square of welding current:

$$Q = I^2Rt$$

Q = amount of heat generated during the welding process
I = welding current applied
R = electrical resistance setup at interface of metal sheets
t = time for which current is applied

Electrodes are generally cone-shaped, so to concentrate the current into a small portion of the workpiece and here, where the temperature is the highest, the fusion takes place (weld nugget spot). The electrodes clump the sheets to hold them together through mechanical force avoiding misalignments and are built in copper alloy to ensure heat generation on the workpiece rather than on the electrodes, thanks to its high thermal conductivity and low electrical resistance.

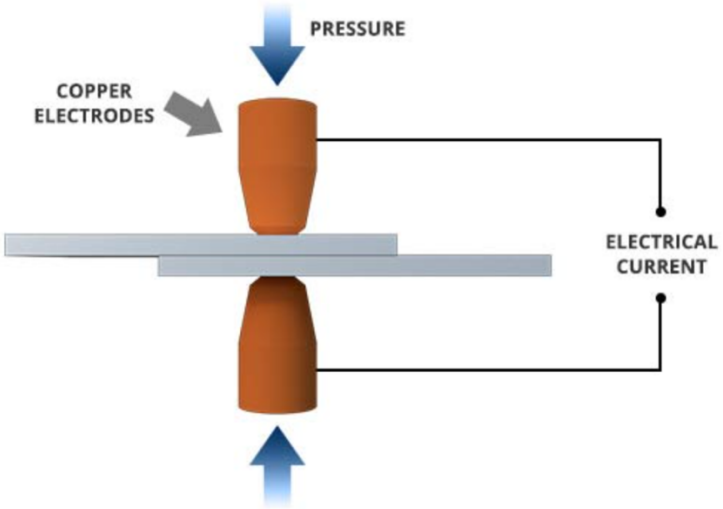

Figure 4.2 - Resistance Spot Welding (RSW) [196]

RSW process takes place into 2 steps:

1. First, electrodes are positioned on the surface of the sheets to be welded, and pressure is applied.
2. Then, welding current is applied from electrodes for a small amount of time and when current is turned off, pressure is kept for the welded material to solidify by cooling down.

Based on the timespan of welding current application, the value of current applied, and the resistance between the two electrodes and between metal sheets and electrodes, it is possible to regulate the amount of heat generated during the process, on the basis of the specific material characteristics of the sheets to be welded.

Usually, this type of welding is preferably applied to sheets of steel or titanium, because of their low thermal conductivity and high electrical resistance which allow for an easy spot weld. The process can be also applied to other materials, like aluminum and magnesium, but

for the former larger welding currents are required (due to higher thermal and electrical conductivity), and for both of them oxide might form on the surfaces due to their high thermal conductivities.

RSW process is largely employed in automotive manufacturing and aerospace industry, but also in other areas of application like electronics, home appliances, medicine (e.g., orthodontist sector), construction, railways, and other fields for the production of thin sheets components.
In the manufacturing of a car, around 2000 or 3000 welding spot points can be implied and one of the reasons why this welding type is so widely employed is related to the easiness for it to be automated with robots and innovative technological systems, which makes it appropriate for manufacturing lines producing large quantities (e.g., carmakers).

Resistance seam welding (RSEW) is a variant of spot welding which uses turning wheel-shaped electrodes and the workpiece slides between such electrodes while welding current is applied, so that a continuous seam between the two parts is created [197].

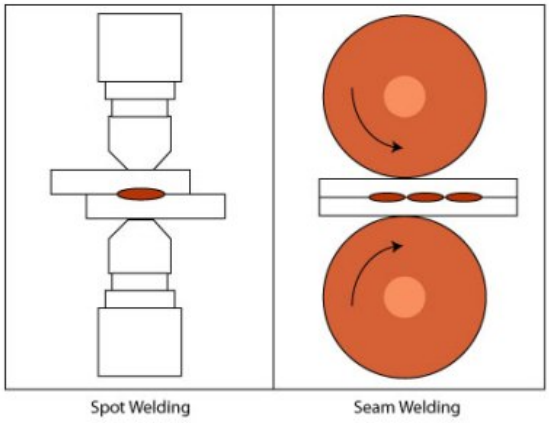

Figure 4.3 - Resistance Spot Welding vs Resistance Seam welding [198]

4.2 Arc welding

Arc welding process [199] [200] [201] [202] [203] is a fusion welding process used to join metal pieces by melting them with an electric arc, that is a controlled electrical discharge between the electrode and the base material to be welded. The arc is generated through electricity from a power source (either direct or alternating) and the welding region is protected from atmospheric contamination by the establishment of a shielding gaseous conductive medium (called arc plasma). The arc generates an intense and concentrated heat able to melt the materials (with or without filler material), and a strong welded joint is formed as they cool down.
The process can be performed either manually by an operator moving the arc along the joining line or mechanically.

There are many different typologies of arc welding processes, and they are grouped into 2 main categories: consumable electrode methods and non-consumable electrode methods,

depending on whether the electrode itself melts and becomes part of the weld seam or not, by simply behaving as arc conductor without melting.

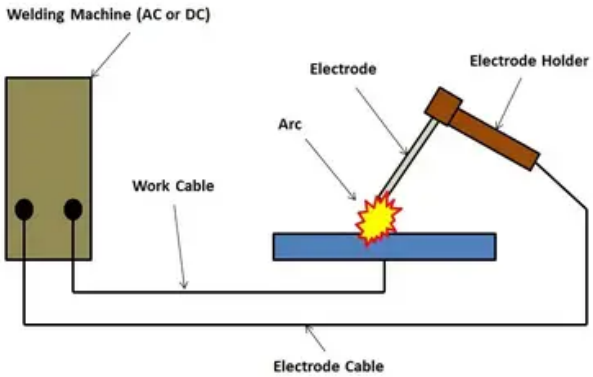

Figure 4.4 - Arc welding [203]

4.3 Friction Stir Welding (FSW)

Friction stir welding (FSW) [204] is a solid-state welding process which joins materials thanks to a non-consumable rotating tool which generates heat.
Indeed, pieces are joined in the solid phase through a rotating tool composed of a plane or shaped base, called shoulder, and a tip, called pin. The tool is rotated and plunged along the joint line at the interface between the pieces to be joined. Due to friction of the tool at the materials surface, heat is released and this, combined with the force applied, allows for plastic deformation of the materials that are joined through generation of mixed softened metal, which creates the weld seam as it solidifies back [205]. To create the solid weld, additional plasticized material can be contained in the shoulder and injected by the tool to mix with the softened material in the weld seam generation. The procedure is defined as solid-state since the material is not completely melted: it reaches a certain ductility level which allows the tool to progress forward along the interface and mix the two materials, which might also be dissimilar [206].

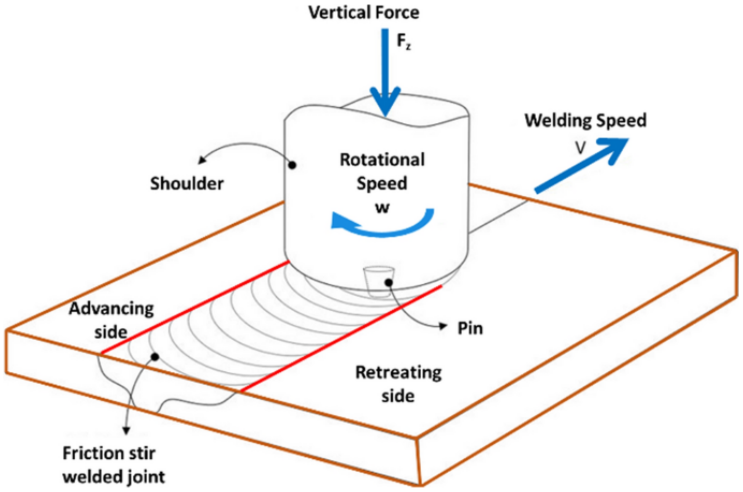

Figure 4.5 - Friction Stir Welding [207]

4.4 Laser welding

Laser welding is a precise process used to join materials (mainly metals or thermoplastics) through a laser light beam for the creation of the weld. The laser beam is directed towards the workpieces and melts them, so that the softened pool cools down and solidifies by fusing the pieces together [208] [209].

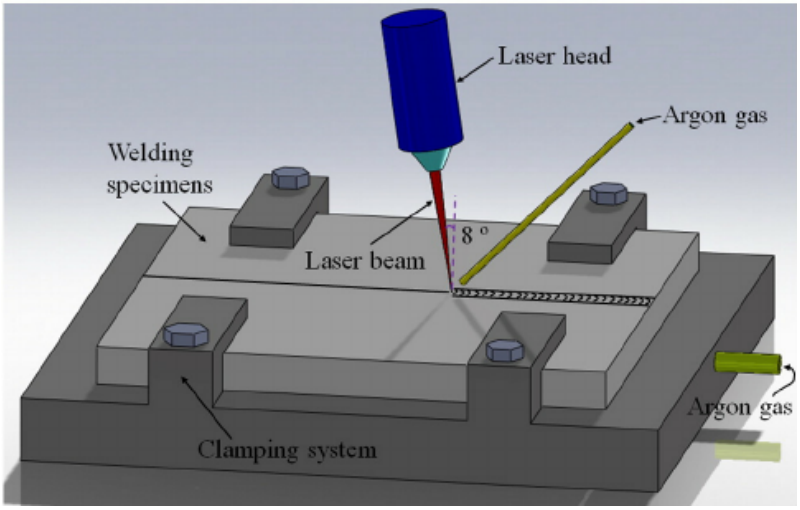

Figure 4.6 - Laser welding [210]

The laser welding process brings some benefits with respect to the conventional welding methods [211] [212]:

- High precision and speed. The laser beam can be controlled very precisely, by directing the energy exactly to the location where the weld needs to occur. Also, it can proceed fast along the joint surface (especially for thinner materials) thanks to the high heat generated.
- Lower distortions. The consequence to the previous characteristic is the minimization of the workpiece zone affected by the welding heat in a way to keep distortions to the minimum. Being the energy beam very concentrated, it heats and melts the welding pool very quickly, so that the laser can fast proceed along the weld seam and the generated heat has no time to spread through the material as it happens for other welding procedures.
- Deeper penetration. Since a highly concentrated heat is generated, laser welding allows to generate narrow and deep welds between thicker materials.
- Adaptability. Laser welding process is versatile as it adapts to joining several typologies of materials, like metals, plastics, and some ceramics. Also, dissimilar materials can be welded through laser welding, which is usually not possible with other welding methodologies.
- High-quality results in terms of strength and seam appearance. The resulting welds are of high quality in terms of mechanical properties and free of some usual weld defects, and the generated seam will match the strength, durability, and corrosion resistance properties of the welded materials.

- Higher productivity. This advantage is related to high precision and speed characteristics of the process, which allow to deliver high-volume production combined with high-quality production, and to the possibility for laser welding process to be easily automated.

5. DT welding applications – Literature review

5.1 Research procedure

In the following, a description of the research procedure applied to derive the information used in this thesis work is presented.

Firstly, in order to start developing some personal knowledge on the topics, the necessary definitions (e.g., of digital twin and machine learning technologies) were found on the web, by looking for the most reliable, detailed, and trusted websites (e.g., IBM, Siemens, etc.).

After that, the research moved down to a more detailed level thanks to Scopus instrument [213], and a top-down research approach was carried out, starting from high-level keywords down to more precise and detailed queries.
So, at first the research was carried out more generally to have a wide overview of the number of articles existing in terms of Digital Twin, Machine Learning, ANNs, and CNNs. Each keyword was used to search within 'Article title, Abstract, and Keywords' of the research papers, and the keywords have been then merged among themselves and with welding-related keywords to obtain more focused results.
The results of this preliminary research process are displayed in a Venn diagram and in a table which allow to have a first overview of the number of articles appeared for each keyword used in the procedure and by merging such keywords. These results have been regularly updated during the thesis work development, since the number of works found happened to change much from time to time, and the reported numbers date back to October 5th, 2023 (last update).

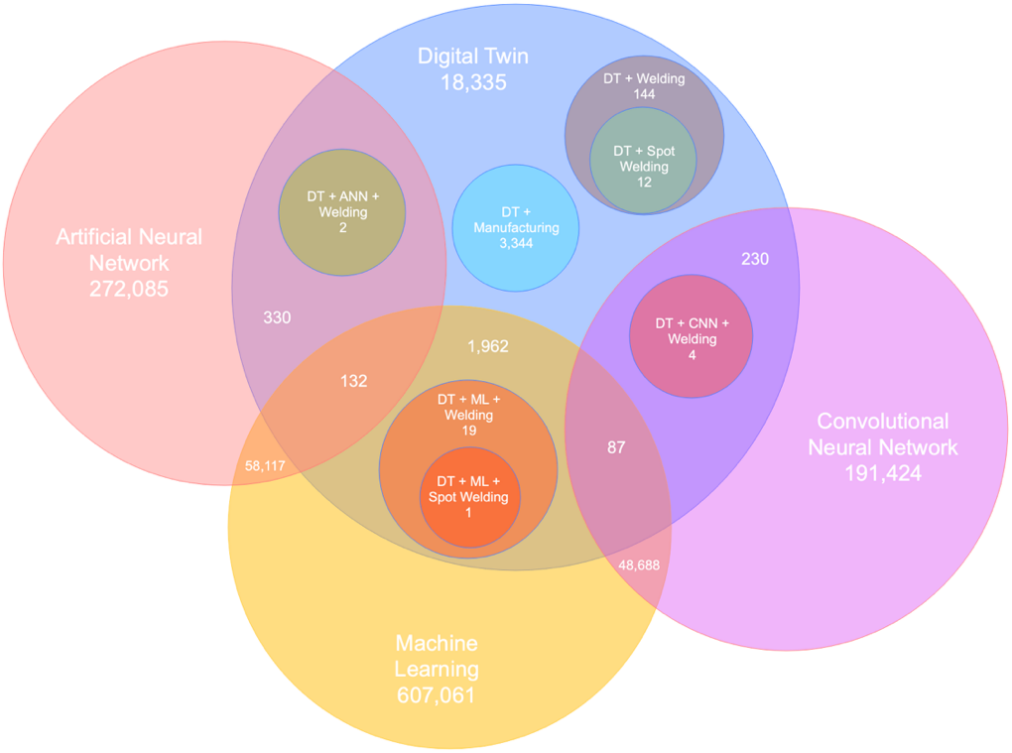

Figure 5.1 - Research process Venn diagram

Keywords	Number of articles	Peak year(s)	Main countries	Main subject areas
DIGITAL TWIN	18,335	2022	China, Germany, US	Engineering, Computer science
DIGITAL TWIN + MANUFACTURING PROCESS	3,344	2022	China, Germany, US	Engineering, Computer science
DIGITAL TWIN + WELDING	144	2022	China, Germany, Sweden	Engineering, Computer science
DIGITAL TWIN + SPOT + WELDING	12	2022	Sweden, Germany, China	Engineering, Computer science
MACHINE LEARNING	607,061	2022	US, China, India	Computer science, Engineering
MACHINE LEARNING + DIGITAL TWIN	1,962	2022	US, China, Germany	Engineering, Computer science
MACHINE LEARNING + DIGITAL TWIN + WELDING	19	2021	US, China, France, Sweden	Engineering, Computer science
MACHINE LEARNING + DIGITAL TWIN + SPOT + WELDING	1	2018	Sweden	Engineering
ARTIFICIAL NEURAL NETWORK	260,503	2022	China, US, India	Computer science, Engineering
ARTIFICIAL NEURAL NETWORK + DIGITAL TWIN	330	2022	China, Germany, US	Engineering, Computer science
ARTIFICIAL NEURAL NETWORK + DIGITAL TWIN + WELDING	2	2023	China	Engineering, Materials science, Physics and Astronomy
CONVOLUTIONAL NEURAL NETWORK	177,565	2022	China, US, India	Computer science, Engineering
CONVOLUTIONAL NEURAL NETWORK + DIGITAL TWIN	230	2022	China, US, Germany, UK	Engineering, Computer science
CONVOLUTIONAL NEURAL NETWORK + DIGITAL TWIN + WELDING	4	2021	US, China	Engineering, Computer science, Chemical engineering, Physics and Astronomy

Table 5.1 - Research process table

In the following, a description of the steps followed in the research process is reported:

a. ‘Digital twin’ broad search

Figure 5.2 – Digital twin search, Scopus keywords

Clearly, such a general search led to as much as 16,113 articles resulting and by looking at documents released per year, the number of articles increased much from 2014 onwards, till reaching its peak in 2022. This reflects the increasing interest that digital twin is rising among researchers, since the number of articles on this topic have been exponentially growing over the last few years and the trend will probably be confirmed as 2023 comes to an end.
Most of the articles resulting from the search are out of topic since just mentioning the digital twin technology without actually developing it (especially among those dating back to before 2018), while most of the ones that are precisely focused on digital twin were developed between 2020 and 2023.
In addition, most of the documents found are from China, and Germany and US follow.
For what concerns the application field, documents related to digital twin are mainly concerning the engineering area, followed by computer science.

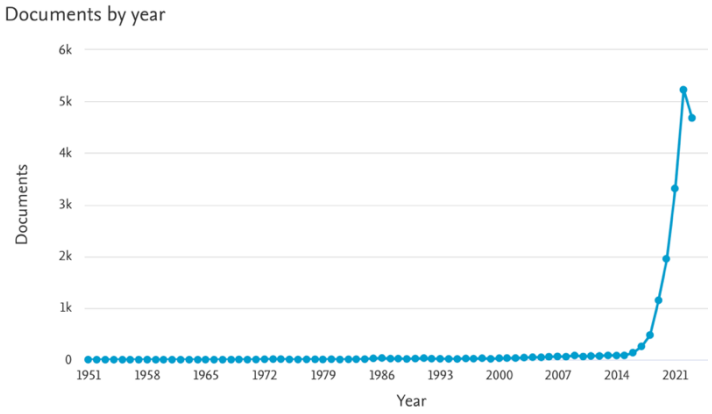

Figure 5.3 - Digital twin works by year [213]

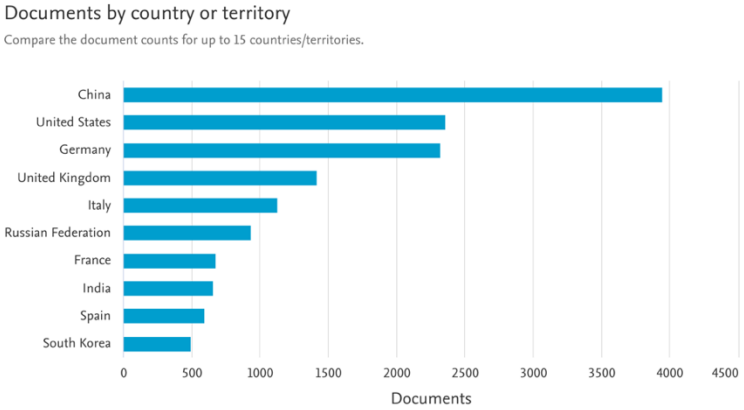

Figure 5.4 - Digital twin works by country [213]

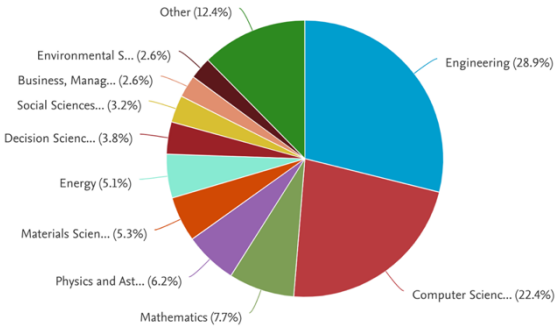

Figure 5.5 - Digital twin works by subject area [213]

b. ‘Machine learning’ broad search

Figure 5.6 - Machine Learning search, keywords

This research led to a much larger number of results than for digital twin, indeed 607,061 articles were derived by using ‘Machine Learning’ as a keyword.
The number of documents per year started increasing from around 2003 on, that is much earlier than in the case of digital twin, and the peak occurred in 2022 as well.
Most of the works are developed in US, followed by China and India, and they mostly cover computer science and engineering fields.

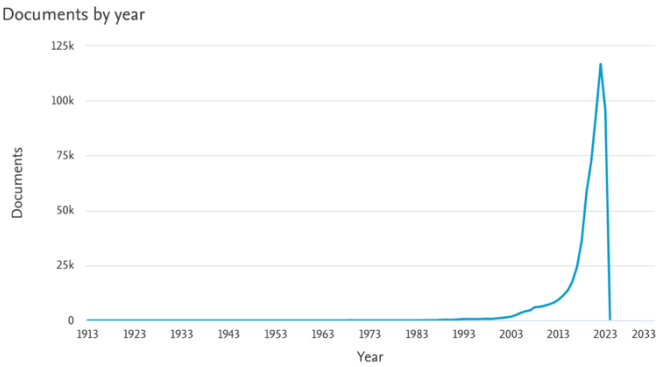

Figure 5.7 - Machine Learning works by year [213]

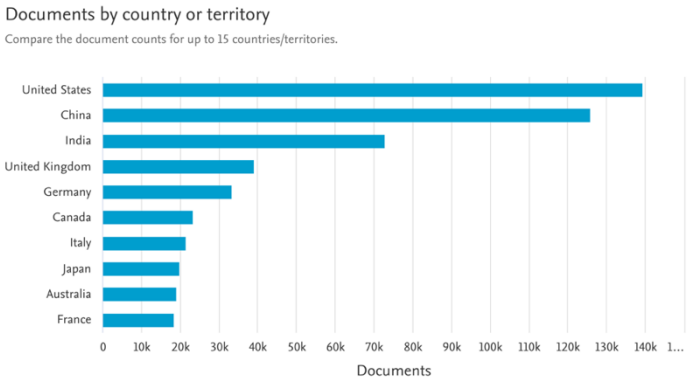

Figure 5.8 - Machine Learning works by country [213]

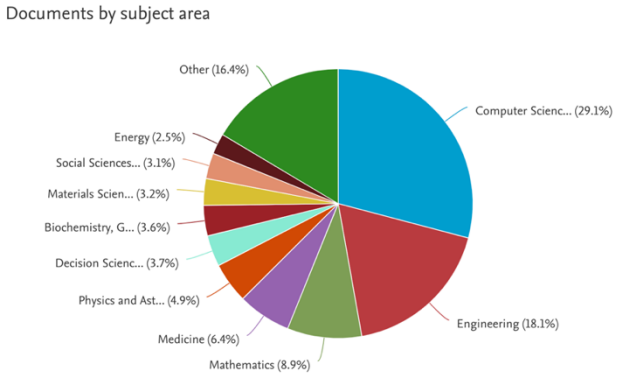

Figure 5.9 - Machine Learning works by subject area [213]

c. A more specific search was carried out to find out whether there are any research works applying artificial neural networks and convolutional neural networks in a digital twin framework, and with a focus on welding processes. A good number of articles resulted by searching for ANN and CNN applications within general digital twin case studies, while very few when zooming in on welding-focused case studies (2 ANN paper works and 4 CNN paper works).

The above-described procedure was carried out so to have a general overview of the typologies and yearly/geographical distribution of existing works focused on digital twin and machine learning technologies. In order to deepen the research and get results that are more centered around the topic of interest (i.e., digital twin applications to welding processes), a search has been performed by inserting 'digital twin' (specifically searched within 'Article title' category) + 'welding' keywords on Scopus search instrument. As expected, the number of resulting articles is strongly reduced down to a reasonable number for the analysis purposes of this thesis work (60 as for October 5th, 2023) and some of these paper works have been analyzed from the most recent down to the least recent ones, discarding those that are not interesting for the purposes of this thesis work. Different kinds of digital twin applications have been analyzed, from those applying machine learning and neural networks, to those simply using a simulation software and displaying a dashboard to the operator.

5.2 Research results

Following this reasoning, out of the 60 works resulting, 21 have been identified as the most interesting for this research and analyzed to understand how digital twin has been adopted over the years and which are the main applications purposes.
After the analysis, the various articles and paper works have been categorized on the basis of the type, uses, and purposes of each specific application analyzed (e.g., real-time monitoring, anomalies detection, etc.). In the first table below, a description of the categories used to classify the digital twin applications is presented, while in the second table the actual classification of the different case studies is performed.
As it will be clear from the analysis, almost all digital twin applications analyzed allow for some kind of real-time monitoring of the process current condition. This is in line with the cornerstone of the digital twin technology, that is the one of providing for real-time mapping, mirroring, and synchronization between virtual and real environments.

Real-time monitoring	Digital twins allowing the operators to monitor the mirrored physical entity. Examples collected from the analyzed articles: • Dashboard for real-time monitoring of parameters regarding the process and its performance (e.g., temperature transient field) • Real-time 3D monitoring of the system through 3D modeling software • Real-time quality monitoring thanks to the outcomes of a simulation* • Real-time monitoring of the process state (e.g., job, idle, etc.) with real-time mapping and synchronization between virtual and real process states • Real-time quality inspection and monitoring by extracting 3D scans of the parts • Real-time quality monitoring through a CNN recognizing patterns in the process control charts • Real-time monitoring of the process through VR tools • Real-time monitoring of human operators health condition by having them wear AR and IoT devices
Simulation*	Digital twins embedding a simulation model to carry out either offline simulations or real-time simulations. Examples collected from the analyzed articles: • Offline simulation to verify the results of an optimization or anomalies detection algorithm, before intervening in the real environment • Offline simulation to generate or integrate the input dataset needed to train a ML algorithm • Real-time simulation fed with real-time data (e.g., 3D scan of the parts) and able to simulate the corresponding final quality, so to adjust process parameters accordingly • Offline non-nominal simulation, fed with 3D scan of the parts to be welded • Offline simulation of process tools configuration to help in process design • Simulation within an AR environment to visualize possible configurations for a monitoring system to be established
Anomalies or defects detection	Digital twins adopting (ML) algorithms or computational models able to identify, and even classify, anomalies in the process and defects in the workpieces. Examples collected from the analyzed articles: • Detection of weld incompleteness through a deep learning algorithm • Real-time detection of welding flaws, quality issues, and anomalies in the process state through a ML algorithm (e.g., CNN able to recognize unusual patterns in the control charts of the process)
Prediction	Digital twins involving machine learning models so to have predictive capabilities. Examples collected from the analyzed articles: • Prediction of real-time temperature field given welding process parameters, some real-time extracted through sensors (e.g., through ML algorithms or an iterative numerical computation model) • Prediction of future defects with a real-time analysis of process parameters • Classification of the human operators' professional skill level through motion tracking and ML algorithms • Predictions by ML algorithms trained in a transfer learning approach over the different phases of the product or process lifecycle

Optimization	Digital twins including algorithms for some kind of optimization. Examples collected from the analyzed articles: • Optimization of the welding robot starting position and the path followed by the robot to perform the weld through advanced algorithms • Optimization of the line scheduling sequence through simulation* and algorithms • Optimization algorithm for the design of process tools that can ensure final quality • Optimization of the workshop layout through algorithms fed with real-time acquired process data
Decision-making support	Digital twins providing the operator with suggestions about how to operate. Examples collected from the analyzed articles: • Suggestions about how to proceed in case of welding defects (identified through a CNN) • Suggestions about how to adjust the process (e.g., by adjusting fixtures) in a way to improve final quality (ML algorithm) • Suggestions through an AR environment about the optimal configuration of the process monitoring system • Suggestion of the optimal workshop layout • Suggestions about how to optimize the health and operating conditions of human workers

Table 5.2 - Identified categories for DT applications

Case study	Article Title	DT category	Purpose	Welding process	Machine Learning-based
1 [76]	'Real-time temperature monitoring of weld interface using a digital twin approach'	• Real-time monitoring • Prediction	Real-time temperature monitoring at the welding interface and prediction of the materials melting instant through a machine-learning algorithm. The dataset needed for the model was generated by a finite element model able to estimate the temperatures at the welding zone corresponding to different applied welding currents.	Special type friction process for dissimilar materials	A linear regression model is trained over a dataset mapping input current to temperature at the welding zone. After training, the algorithm is able to predict the interface temperature on the basis of real data about current applied.
2 [214]	'Digital Twin Implementation of Autonomous Planning Arc Welding Robot System'	• Real-time monitoring • Optimization • Simulation	Real-time 3D monitoring of welding robots, optimization algorithm to identify a collision-free path for the robots to perform the weld, and validation of the robot path through offline simulation before transferring the plan to the physical process.	Arc welding	-
3 [75]	'Real-time detection method for welding parts completeness based on improved YOLOX in a digital twin environment'	• Simulation • Anomalies or defects detection	3D simulation to virtually generate the dataset (of images and videos) needed for the training a deep learning algorithm to detect parts incompleteness prior welding.	Arc welding	A deep learning algorithm is trained over a hybrid dataset of components' images and videos (both from the real and the virtual environments) and then receives real-time visual information as input to detect weld incompleteness defects.

4 [81]	'Quality Monitoring of Resistance Spot Welding Based on a Digital Twin'	• Real-time monitoring • Simulation	Real-time collection of welding current, voltage, and electrode force data to feed the real-time welding simulation whose results allow to monitor welding quality. In case of missing compliance with quality requirements, welding parameters are adjusted in real-time accordingly.	Resistance Spot Welding	-
5 [14]	'Deep Learning-Empowered Digital Twin Using Acoustic Signal for Welding Quality Inspection'	• Real-time monitoring • Anomalies or defects detection	Real-time collection of acoustic signal data from the real process for flaws detection through a deep learning algorithm, so to monitor and ensure welding quality.	Arc welding	A deep-learning algorithm based on a convolutional neural network, receives sound signals data as input and is able to classify them based on their time and frequency features, so to detect any welding faults affecting final quality.
6 [5]	'A Novel Method of Digital Twin-Based Manufacturing Process State Modeling and Incremental Anomaly Detection'	• Real-time monitoring • Anomalies or defects detection	Real-time identification of the process state (through a hierarchical finite state machine method) and state synchronization between virtual and real models. Identification of anomalies in the process which can affect the assembly final quality.	Arc welding	A machine learning algorithm carries out a detection mechanism able to identify abnormal behavior by the equipment, on the basis of the real process state.
7 [215]	'Digital twin model-driven capacity evaluation and scheduling optimization for ship welding production line'	• Real-time monitoring • Simulation • Prediction • Optimization	3D simulation model for real-time optimization of the line scheduling sequence: it is fed with real-time welding data (e.g., current, temperature, voltage, etc.) and with the help of data mining and intelligent algorithms it is able to assess parameters affecting production capacity and optimize process variables accordingly. Real-time analysis of welding parameters to predict possible defects in advance (i.e., to perform welding quality prediction) and adjust parameters consistently. Displaying of a dashboard to the operator showing production process real-time parameters (e.g., current, voltage, etc.) and information for process performance analysis (charts showing failure rate, bottleneck, processing time, etc.).	Spot welding	-
8 [211]	'Integration of Industry 5.0 requirements in digital twin-supported manufacturing process selection: a framework'	• Real-time monitoring • Optimization • Decision-making support	Support in solving the process selection problem among different setup alternatives, on the basis of both Industry 4.0 and Industry 5.0 criteria. During operations, to embed human inclusion factors, workers can wear IoT and AR wearables to monitor their physical status and health: with this information, the digital twin can real-time monitor and optimize workers efficiency by giving suggestions and instructions.	Laser welding	-
9 [4]	'Process Simulation and Optimization of Arc Welding Robot Workstation Based on Digital Twin'	• Simulation • Real-time monitoring • Optimization	Identification of the optimal initial position for the welding robot and robot welding path so to improve process efficiency and reduce process time. The process can be monitored, and welding data visualized through an UI interface. A virtual simulation can be performed to test the process before it starts.	Arc welding	-

10 [15]	'A detection and configuration method for welding completeness in the automotive body-in-white panel based on digital twin'	• Anomalies or defects detection • Decision-making support	Immersion of the operator into a MR environment (head-mounted devices) with which they can interact through gestures or voice. Images can be captured through the MR tools and sent to a deep learning algorithm for incompleteness and defects detection. The MR environment will then guide the human user in addressing the defective workpieces (e.g., repair procedures).	Arc welding	A deep learning algorithm based on a Convolutional Neural Network is used to inspect workpieces images coming from the real environment so to detect welding completeness and give configuration suggestions.
11 [192]	'Predicting Geometrical Variation in Fabricated Assemblies Using a Digital Twin Approach Including a Novel Non-Nominal Welding Simulation'	• Real-time monitoring • Simulation	Improving the geometrical quality of the final welding assembly (geometry assurance) and allow for mass customization. Before the welding process takes place, the parts can be 3D scanned to understand their initial (non-nominal) geometrical variation and used to carry out a non-nominal simulation to monitor and predict the quality of the final welding so to adapt the real process settings in a way to optimize the final assembly geometrical quality.	Laser welding	-
12 [212]	'A Survey of Process Monitoring Using Computer-Aided Inspection in Laser-Welded Blanks of Light Metals Based on the Digital Twins Concept'	• Real-time monitoring	Quality automated inspection: the 3D scan of the part is extracted, and quality control is performed through an algorithm which compares the nominal CAD model of the part with the scanned model, to identify any deviations from the design. Such an approach could help in obtaining product licenses by verifying its quality.	Laser welding	-
13 [216]	'Digital Twin for the Transient Temperature Prediction During Coaxial One-Side Resistance Spot Welding of Al5052/CFRP'	• Real-time monitoring • Prediction	Prediction of the transient temperature field generated during the welding process. A machine learning interpolation algorithm is fed with the welding process parameters. To train the algorithm, a FEM model is used to generate additional datapoints to increment the dataset mapping welding process parameters to the transient temperature field.	Resistance spot welding	A machine learning algorithm receives as input a set containing the specific welding process parameters (e.g., welding time, current, force applied, etc.): if such exact data point is mapped already by the existing dataset, then the prediction is easily provided; if this is not the case, then a multiple dimension interpolation is carried out over existing datapoints within the dataset to perform the prediction.
14 [217]	'Integration of Digital Twin and Machine Learning for Geometric Feature Online Inspection System'	• Real-time monitoring • Simulation • Anomalies or defects detection • Decision-making suggestions	Achievement of high welding quality: a CNN is used to identify anomalies by performing pattern recognition over the process control charts. Then, a machine learning algorithm predicts and suggest fixture adjustments to address any anomaly, and such suggestions are iteratively simulated within the digital twin until an optimal decision is conveyed.	Arc welding	A CNN model is able to categorize control chart patterns. The dataset needed for the training process is integrated with data points derived through Monte Carlo simulation. Then, another machine learning algorithm is able to suggest

					adjustments to the welding process to reach the optimal setting.
15 [205]	'Digital Twins in the Design of Tools for Friction Stir Welding'	• Simulation • Optimization	Reduction of costs and complexity of tools and process design. A digital twin model of the tool takes into consideration all the elements that will affect the final weld quality (e.g., welding parameters and operation conditions, etc.) and, through an algorithm and a simulation, it determines the optimal combination of operational parameters and tool configuration to ensure outcome quality.	Friction stir welding	-
16 [73]	'Digital twin modeling for temperature field during friction stir welding'	• Real-time monitoring • Prediction	Real-time monitoring of the 3D temperature distribution around the welding zone so to adjust the welding parameters as the process moves on. This is done by synchronizing the real process with an iterative numerical model, which is able to reproduce the 3D temperature field in real time by applying a heat transfer analysis and fusing real data collected through sensors and data derived through the computational model.	Friction stir welding	-
17 [13]	'Digital Twin for Human-Robot Interactive Welding and Welder Behavior Analysis'	• Real-time monitoring • Prediction	Enhancement of human-robot interaction to increase operational productivity. By using motion tracking, the human welder demonstrates the welding operations and is recorded; a robot executes the demonstrated operation; a digital twin based on virtual reality collects real-time data through sensors and allows for a virtual replica of this interaction. The human operator can visualize the physical process and its performance in real-time through VR glasses. The digital twin includes a machine learning classification model to classify the human welders on the basis of their professional skills.	Arc welding	An SVM machine learning algorithm performs welders' behavior analysis by classifying the professional level of human welders on the basis of the demonstrated welding operation. The algorithm receives data about the movements followed by the welder when demonstrating the welding operation and is able to distinguish between skilled and unskilled welders on the basis of the difference between their operating patterns.
18 [77]	'Towards a digital twin setup for individualized production of fabricated components'	• Real-time monitoring • Simulation	Advancement towards individualized production, where the production process is adapted on the basis of the specific individual welding, by taking into consideration the initial geometrical variations in the parts vs their nominal design, which could affect final quality. 3D scans of the parts are collected before the process takes place to extract their unique features, and real-time fed to a simulation that is run iteratively to predict the final quality of the weld so to adjust the process accordingly and achieve higher final quality.	Laser welding	-
19 [218]	'An AR based Digital Twin for Laser based manufacturing	• Simulation • Decision-making support	Give support in the choice of the process optimal monitoring system to be introduced. Augmented reality is introduced within a digital twin to support the choice of the optimal equipment configuration of the	Laser welding	-

	process monitoring'		monitoring system. The main components of such monitoring system are modeled, simulated, and projected through AR to the user. The model suggests alternative solutions, the user can try to install some components in the AR environment, simulate them, and see the properties of such a configuration.		
20 [74]	'Transfer learning as an enabler of the intelligent digital twin'	• Simulation • Prediction	Application of the concept of transfer learning to the industrial processes. Through transfer learning, machine learning algorithms can learn new tasks on the basis of knowledge acquired on previous related tasks, so to address the complexity of collecting large and diversified datasets for the ML training process and achieve high-quality algorithms. Introducing cross-phase industrial transfer learning allows to transfer knowledge from one asset's lifecycle phase to another and reduce the time needed for a ML algorithm to be trained.		With a digital twin model which includes simulations, machine learning algorithms can be pre-trained and tested even before the actual physical process exists (i.e., during its design phase). As the real system is built, then the algorithm's knowledge is transferred to the real asset (i.e., from design to operation phase) and the network can be fine-tuned through training over the real data available.
21 [219]	'A digital twin-based layout optimization method for discrete manufacturing workshop '	• Optimization • Simulation • Decision-making support	Sub-optimal workshop layout might affect production efficiency and quality due to issues like unclear partitioning, unreasonable distribution of equipment, and unreasonable material distribution route. The optimal layout can be derived through a digital twin which allows to make equipment layout decisions in real-time by collecting data from the process: optimization algorithms are run and iteratively tested through simulations until the optimum is found and the real assets are arranged accordingly.		-

Table 5.3 - Categorization of the analyzed research papers

5.3 Intelligent digital twin - Potentiality of ML in the DT framework

In the previous chapters, digital twin and machine learning have been separately described. As it emerged from the literature review, though, their simultaneous adoption through a Machine Learning-based Digital Twin model, or 'Intelligent Digital Twin' [74], provides several additional benefits with respect to other kinds of digital twins. Indeed, an Intelligent Digital Twin extends the plain digital twin technology by enriching it with the adoption of artificial intelligence, which provides for new capabilities and applications. Such a digital twin model is able to observe the environment through sensor operating data and analyze it with the other models included within the digital twin, in a way to learn new information and gain new knowledge so to carry out tasks like fault prediction, anomaly detection, machine condition evaluation, production sequence optimization, etc.

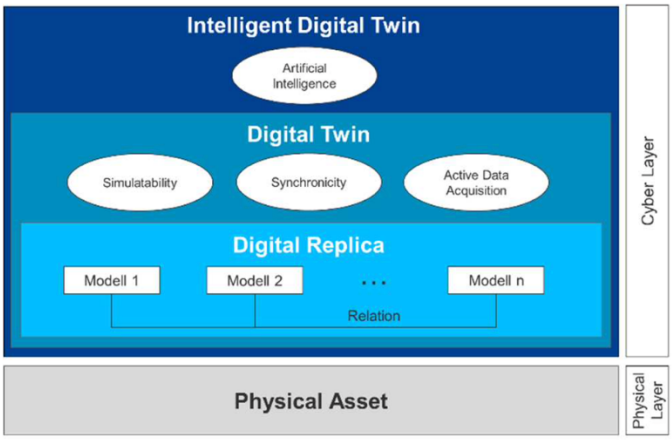

Figure 5.10 - Intelligent Digital Twin vs plain Digital Twin [74]

Some of the main additional capabilities that can be provided by an 'Intelligent Digital Twin' are listed in the following:

- Improved predictive capabilities. Machine learning algorithms are able to analyze historical data and patterns (by training over large and complex datasets) to derive accurate predictions about the future behavior and performance of the physical entity mirrored by the digital twin.
- Anomaly or fault detection. Machine learning algorithms are able to detect any anomaly or fault as deviations appear from the expected behavior and process parameters. As a consequence, such a digital twin could show the operator an alert signal and even suggest possible corrective actions.
- Real-time optimization. As the manufacturing process moves forward, a machine learning model can suggest optimizing actions to the operator on the basis of real-time collected information, so to maximize the manufacturing operation by adjusting process parameters accordingly (e.g., with the purpose of optimizing the final quality).
- Enhanced decision-making support. Thanks to machine learning algorithms, operators have the possibility to make more informed decisions, since the digital twin model can provide more comprehensive and real-time data-driven insights and recommendations to the users. In case the framework implied more advanced control strategies, the digital twin model would be able to even make autonomous decisions and actions.
- Real-time adaptation. Machine learning algorithms are continuously trained over new real-time input data, in a continuous learning fashion, in a way that makes the digital twin constantly able to adapt itself to reality by updating its internal models (e.g., simulations) on the basis of the physical environment evolution. This makes it possible for the digital twin to become more and more accurate and effective the more it is used.

6. A CNN for image classification and quality monitoring

6.1 The context: quality in RSW processes

In case unsuitable choices are taken when setting the welding process, some issues might arise on the assembled workpiece, and even cause some damage to the structure of the welded materials by affecting the overall welding quality.
It is crucial to set the welding schedule and parameters in a way that the produced welds are of acceptable quality. It is the manufacturer who should determine the weld quality for the specific welding process, in line with the American national standard according to which acceptable quality is there in case of 'a weld that meets the applicable requirements' [252].

One way for the manufacturer to verify quality is to use process characteristics as indicators of weld quality. The presence of expulsion, for instance, is frequently used as a welding quality indicator. Being an ejection of liquid metal during the welding process, it clearly indicates the potential weakness of the weld performed and can be easily noticed since usually distinctly visible. Expulsion is an example of defect of the welding process, and its presence is undesirable both in terms of appearance and performance of the spot weld since it might affect its strength.
It can be generally identified through visual inspection of the weld, since traces of ejected liquid metal are visible either at electrode-sheet interface or sheet-sheet interface.
So, an expert operator is able to easily perceive an expulsion as it occurs, without specific experience or training needed: however, expulsion can be also detected through sensors, by extracting welding process signals like electrical signals (e.g., power input), mechanical signals (e.g., electrode force), and acoustic signals (e.g., acoustic emissions). These signals undergo an instantaneous and sudden change in their trend (drop or rise) in case expulsion occurs during the welding process.

6.2 What is an expulsion? What does it entail?

Expulsion phenomenon can be frequently observed during resistance spot welding.
In line with ISO 17677-1 (2021) definition, an expulsion (also called splash, spatter, or flash) is the phenomenon of molten metal particles being ejected either (i) between the faying surfaces of the workpieces (i.e., surface of a sheet in contact with the other sheet to which it is to be joined) or (ii) at the contact interface between sheet and electrode during the welding process. In the second case surface quality might be affected (as well as the electrode life), but if the expulsion is limited to the sheet surface weld strength is not affected. The first kind of expulsion, instead, is strongly undesirable since it deeply affects the weld quality due to liquid metal loss from the nugget during welding. The molten metal taking part to the expulsion, indeed, belongs to the material which was destined to the weld nugget to be generated and is instead ejected [220]. In case of heavy expulsion, voids can appear within the nugget so to cause strong degradation of the weld strength.
In line with that, expulsion should be reduced in RSW so to address the problem of eliminating nonconforming and low-quality welds. It is crucial to be able to monitor and control expulsions, so that welding parameters can be changed to reduce the incidence of such defect.

A wrong welding schedule can cause expulsion. It is common practice in industry to apply a set welding parameters close to the limits beyond which expulsion occurs (e.g., extreme welding current) so to achieve large weld nugget size which is important to meet weld quality requirements. However, since such expulsion boundaries for parameters vary depending on the actual welding condition (e.g., electrode wear), it usually happens that weldings are performed under expulsion condition.

The causes of expulsion are disparate and involve electrical, thermal, metallurgical, and mechanical factors. However, the expulsion process can be explained through the interaction of two forces:

- Force supplied by the electrodes.
- Force from the liquid material within the nugget region towards the solid containment surrounding it. This force derives from the pressure that the molten metal is submitted to due to the compressive force between the sheets. Indeed, during resistance spot welding, the sheets are not allowed to expand freely, but are constrained by the electrodes pressure so to maintain electrical and thermal contact at the interfaces and to try to contain the liquid metal.

Particularly, 'expulsion occurs when the force from the liquid nugget onto the solid containment equals or exceeds the effective electrode force' [221].

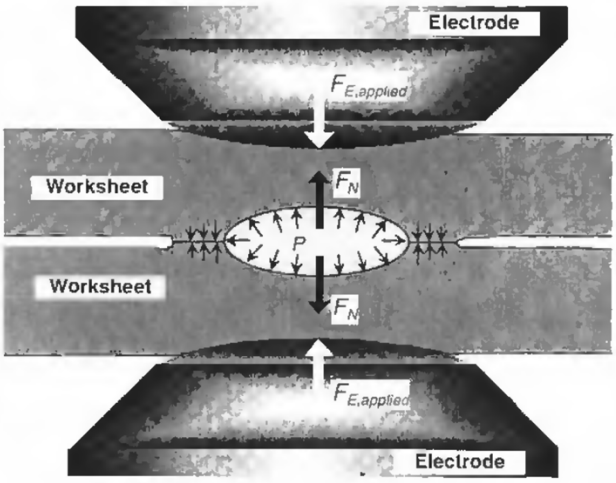

Figure 6.1 - Scheme of the forces acting during resistance spot welding [221]

Depending on the workpieces material, it is possible to understand which force to apply through the electrode by studying the properties of the nugget liquid (e.g., pressure, temperature distribution, etc.) and of the workpiece materials so to predict the severity of the correlated expulsion [221] [222] [223] [220].

6.3 Proposed approach

As mentioned, excessive expulsion is an undesirable outcome, as it is not symptomatic of a quality welding process but is a fallacy of the process itself. Indeed, this unwanted ejection of material beyond the weld zone can result in a weak weld joint (i.e., weak adherence between the material surfaces) and cause quality issues in the next production or operation steps that the workpiece needs to undergo. Structural problems might arise because the molten metal

taking part to the expulsion, belongs to the material which was destined to the weld nugget to be generated. Among other issues, also safety risks and dangers for the surrounding facilities might arise: indeed, molten pieces of hot metal are shot out, and might cause fires, burn the operators' skin, etc.

To these ends, the present work takes part in trying to address the issue of weld joint quality, by being a portion of the overall framework that would be needed to pursue such an objective. A deep learning algorithm, specifically a Convolutional Neural Network model, is proposed to potentially be included within a digital twin framework for welding quality monitoring and control. The inclusion of such an algorithm within a digital twin model could avoid the need for the human operator to perform visual inspection of the joined workpieces for expulsion detection, by instead carrying it out automatically through the image capturing process and the neural network predicting capabilities.
The algorithm performs an image classification task to identify the presence of expulsion or not in the assembled workpiece: data has been acquired from the real environment of a RSW welding process in the form of pictures of the workpieces after the assembly has occurred, and then submitted to the algorithm which is able to extract features from the images themselves and classify them accordingly. Specifically, a simple and explicit variable is extracted, that is, whether the final welded assembly shows any expulsion or not: this variable represents the dependent variable for the CNN model to predict by receiving and being trained over captured images as input data.

As already mentioned, the proposed CNN is not thought as a standalone solution, but as an integral component of a larger digital twin framework which could help in quality monitoring of the process. Indeed, as described within this thesis work, the combination of digital twin technology and advanced machine learning models is proving to have great potentialities in terms of real-time quality assurance, in welding processes as well as in other fields.

In the diagram below, the steps followed for developing the presented application are highlighted, so to simplify the reading of the next paragraphs in which the different phases are described in detail.

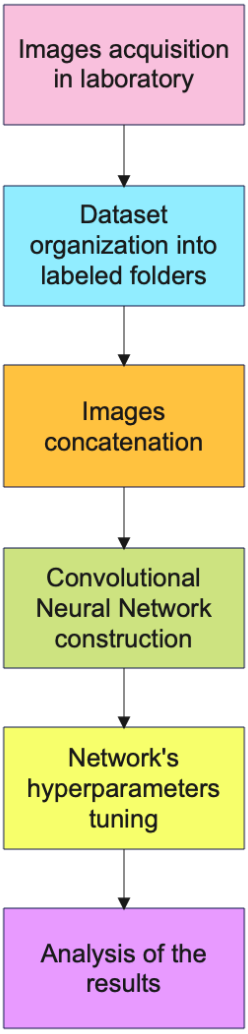

Figure 6.2 - Steps followed to develop the present application

6.4 Experimental campaign in laboratory

For the purpose of gathering images needed to structure the dataset for the algorithm to run, an experimental campaign was carried out in J-Tech laboratory [224] with the help of an expert operator through an industrial 650 kVA Medium Frequency Direct Current (MFDC) resistance spot welding machine with a control unit, mod. TE700 by Tecna industry.

156 DP590 steel slats of size 45x105x1mm were used as base materials for the welding process, so that 78 welds were carried out overall (2 sheets joined per operation). For each weld completed, a picture was taken of front and back of the welded specimens (as shown in *Figures 6.3-6.4* examples) and a table was filled with the operating parameters of the specific welding process and the opinion of the expert operator (over a 0-4 scale) about the existence or not of any expulsion on the workpiece and its severity.

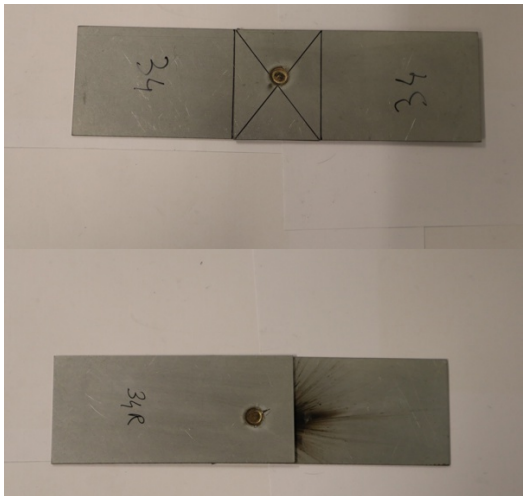

Figure 6.3 - Workpiece with expulsion

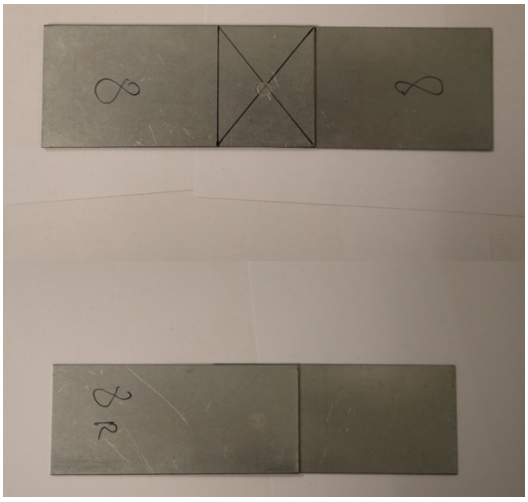

Figure 6.4 - Workpiece without expulsion

6.5 The dataset

As shown in *Figures 6.3-6.4*, completed workpieces are numbered through an ID going from 1 to 78, specifying whether the picture is frontal or back, by adding 'R' to the identification code in case of back image. This step revealed to be pretty useful after the lab experiment when ordering the pictures and structuring the dataset in the best way possible to train the algorithm.
As further explained in the following description of the algorithm, a supervised learning approach is used and for this reason, the dataset elements needed to be ordered and labeled: the set of pictures has been, indeed, split between those workpieces showing expulsion and those not showing any, as shown in *Figure 6.5*.

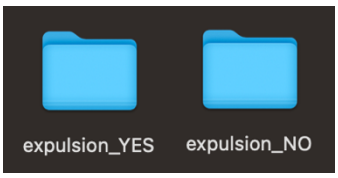

Figure 6.5 - Labeled folders

Within each folder, the images are named after their identification number and the class folder they belong in (e.g., 78_YES, 18_NO, etc.).

6.6 Images concatenation

Right after the lab experiment, there were 156 images overall: 2 pictures for each weld, representing front and back of the welded workpiece (as shown in *Figures 6.6-6.7*).

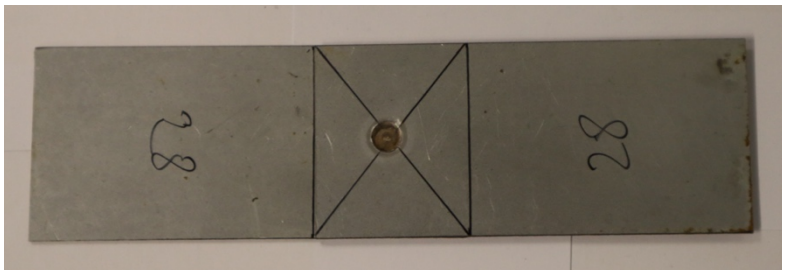

Figure 6.6 - Front picture of the workpiece

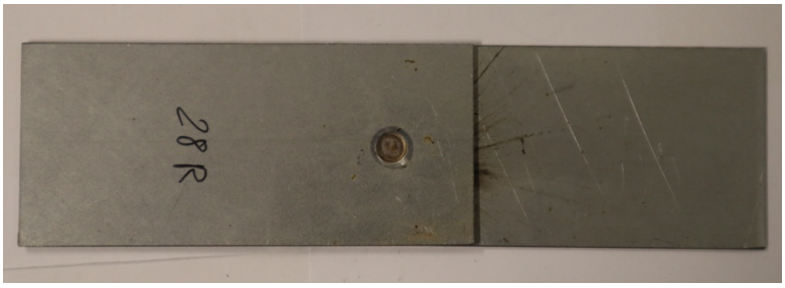

Figure 6.7 - Back picture of the workpiece

The algorithm, though, needed to receive as input one single picture for each welding process so to be able to understand whether that specific process generated any expulsion or not on the single merged workpiece.
For this reason, before writing the actual CNN algorithm, a preliminary code was written and run in order to concatenate front and back pictures for each weld, ending up with one single image for each process, for a total of 78 pictures overall (as the number of total welds performed).

6.6.1 Image concatenation – Code

```python
import os

import PIL
from PIL import Image

def concat(im1, im2):
    newIm = Image.new('RGB', (im1.width, im1.height + im2.height))
    newIm.paste(im1, (0, 0))
    newIm.paste(im2, (0, im1.height))
    return newIm

expulsion_YES = "dataset/expulsion_YES"
expulsion_NO = "dataset/expulsion_NO"

for i in range(78):

    path1='dataset/expulsion_YES/{0}_front.JPG'.format(i+1)
    path2='dataset/expulsion_NO/{0}_front.JPG'.format(i+1)

    if os.path.isfile(path1):
        front=Image.open('dataset/expulsion_YES/{0}_front.JPG'.format(i+1))
        back=Image.open('dataset/expulsion_YES/{0}_back.JPG'.format(i+1))

        newIm = concat(front, back)
        print(newIm)
        newIm = newIm.resize((600,800))
        newIm.save('NewDataset/expulsion_YES/{0}_YES.JPG'.format(i+1), optimize=True, quality=95)

    if os.path.isfile(path2):
        front=Image.open('dataset/expulsion_NO/{0}_front.JPG'.format(i+1))
        back=Image.open('dataset/expulsion_NO/{0}_back.JPG'.format(i+1))

        newIm = concat(front, back)
        print(newIm)
        newIm = newIm.resize((600,800))
        newIm.save('NewDataset/expulsion_NO/{0}_NO.JPG'.format(i+1), optimize=True, quality=95)
```

Figure 6.8 - Screenshot of Images Concatenation code from Jupiter Notebook platform

As a first step, the needed libraries were imported:

a. Python built-in 'os' module, which allows to interact with operating systems and the local directories on the device [225].
b. Pillow (PIL, Python Imaging Library), which is a library providing image processing capabilities and, in particular, the 'PIL.Image' module which supplies the PIL image class and many correlated functions to manage images [226].

A function is defined to carry out the concatenation of front and back images of the individual weld, receiving indeed front and back images as input parameters (JPG images). Through *PIL.Image.new()* function, a new empty Image object (newIm) is constructed with specified size (width of one image and height equal to the sum of heights of the two images). Through *PIL.Image.paste()* method, front and back pictures are pasted to this new Image instance and vertically concatenated one to the other.

A 'for' loop is run 78 times, one per each final image to be obtained. Each image has been accurately named specifying its identification number and whether it is a front or back picture, and then saved within the expulsion_ YES and expulsion_NO folders. This allows to concatenate each frontal image to its corresponding back picture through the previously defined function, resize, and re-save the new image in a new database through the *PIL.Image.save()* method. The size of the concatenated image initially corresponded to (6000, 8000) pixels, and has been proportionally downsized by a x10 factor to (600, 800) for the future network to run more smoothly with input images of manageable size.
The new dataset containing the concatenated images is again structured into expulsion_NO and expulsion_YES folders, as it will be expected by the ML algorithm based on a supervised

learning approach, and each image is named after its identification number and YES/NO expulsion feature.

An example of final image is given in *Figure 6.9*.

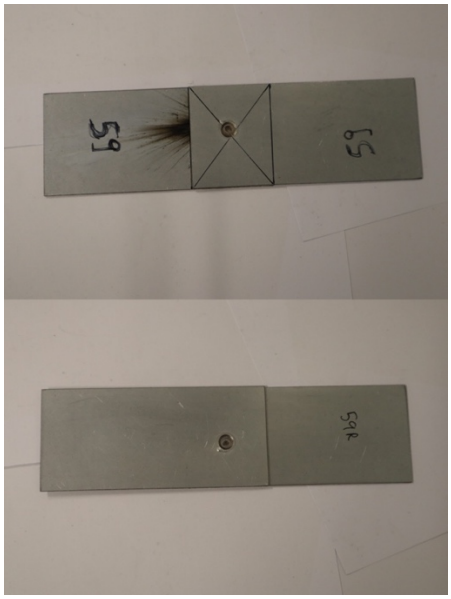

Figure 6.9 - Example of resulting concatenated image (59_YES)

6.7 The ML algorithm – A Convolutional Neural Network

As anticipated in the introduction, the purpose of this case study is the one of developing a machine learning algorithm which is able to learn how to distinguish images representing welded workpieces with expulsion from those without [121].
This objective lies within the image recognition realm and as such, a convolutional neural network has been developed for this purpose, able to get images as input and perform a binary classification of such data inputs: the algorithm is, indeed, able to identify the category each workpiece image belongs to, that is 'expulsion' or 'no expulsion'.

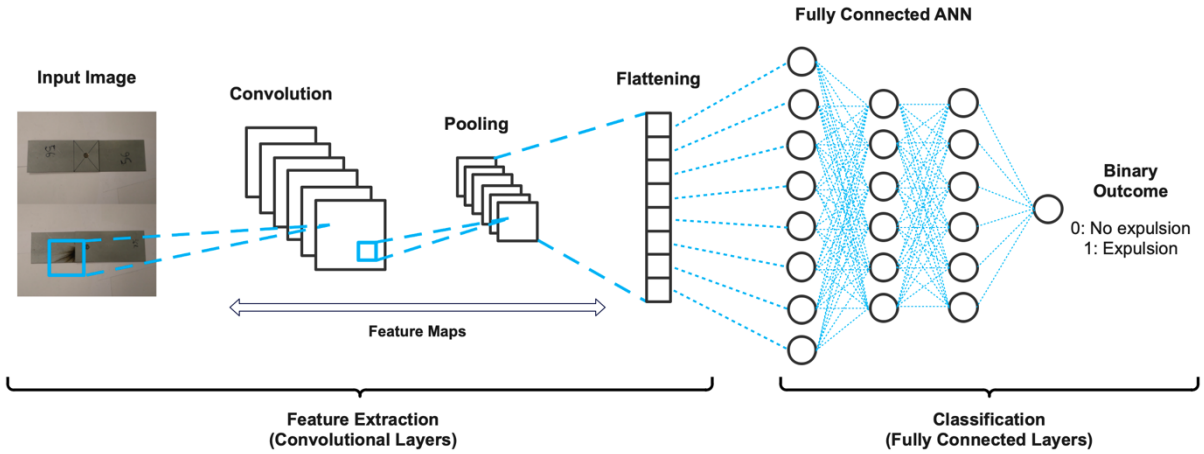

Figure 6.10 - Scheme of the developed CNN for image classification

The developed model is a supervised machine learning algorithm, since the input images are structured into labeled folders for the model to understand the two categories of interest and be able to learn how to distinguish between them through the training process.

In principle, starting from the full dataset, it is possible to manually split it into training, validation, and test sets through the 'simple hold-out split' methodology, by holding out a certain percentage of samples from the dataset to constitute validation and test sets. Then, the algorithm is trained over the selected training set by iteratively changing synapses weights, and its performance is evaluated through the chosen validation set, which contains new data not yet seen by the model.
Still, such splitting approach is not completely unbiased and depending on the selected validation and test sets and, consequently, the remaining training set, the performance metrics for the algorithm might change. For instance, it might be that the selected training set of samples is not representative of the patterns in the real population and so, as the model is applied to real cases and new data, it does not work properly.
So, the issue is related to how to split the full dataset between training, validation, and test sets, so to properly validate and test the model accuracy, with the purpose of deriving the best model configuration (in terms of hyperparameters setting) to be then applied to real cases after the training process has been satisfactorily completed. The most appropriate solution is to apply the so-called K-fold cross-validation technique, which is one of the most used testing methodologies and is based on training and validating the model repeatedly to perform the hyperparameters tuning procedure, till reaching the most adequate model configuration in terms of high prediction accuracy.
First of all, a certain number of samples in the original datasets are set aside for the test set and are not used until the end of the training process. In this specific application, the choice of the images to include within the test set has been done in a way for the test set to be representative of the whole dataset population, so by maintaining the same proportion of images belonging to the first class (i.e., 'expulsion') with respect to those belonging to the second class (i.e., 'no expulsion'). Since the whole dataset contains 48 pictures showing expulsion, and 30 not showing any, then the test set keeps this 5:3 proportion.
Then, the remaining portion of the dataset is split into K subsets of equal size (called 'folds'): at each iteration, one of the K folds is kept aside and used as validation set while the remaining K-1 folds are used as training set to train the model. The performance of the specific model configuration on the validation set is calculated at each iteration, and after all K iterations are completed, the K scores are averaged across all iterations so to obtain the overall performance of the specific model configuration (with a certain hyperparameters setting) [152].
In the present application, the original dataset has been split into training set on one side, and validation and test sets on the other, according to an approximately 7:3 proportion: the set aside test set contains 8 data samples (images), and the training and validation sets contain 56 and 14 samples respectively. In the cross-validation procedure, K is set to 5 meaning that the dataset (after test set exclusion) is split into 5 folds and 5 training iterations are performed, each with a different validation set and a different resulting accuracy value. The model is then evaluated based on the average accuracy over the 5 sets used for validation.

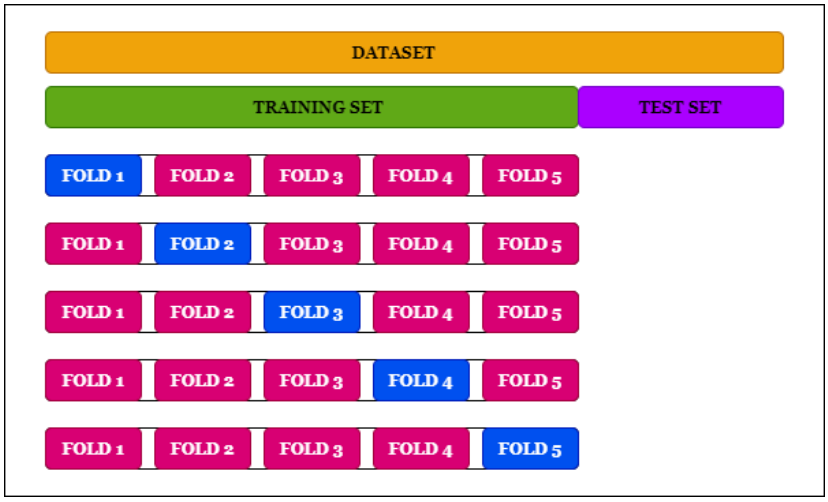

Figure 6.11 - 5-fold cross-validation scheme [227]

In the specific case study object of this thesis work, a special typology of K-fold cross-validation was applied, called Stratified K-Fold Cross Validation, which is more suitable for a binary classification problem, as the one under consideration.

6.7.1 Convolutional Neural Network – Code

In the following, a line-by-line description of the developed code is proposed.
The algorithm was developed in Python coding language on Jupiter Notebook environment [121].

```
1  import tensorflow as tf
2  import tensorflow_datasets as tfds
3
4  from tensorflow import keras
5  from tensorflow.keras.preprocessing import image
6  from tensorflow.keras import layers
7
8  from sklearn.model_selection import StratifiedKFold
9  from sklearn.metrics import confusion_matrix, accuracy_score, log_loss
10 import numpy as np
11 import matplotlib.pyplot as plt
12
13
14 import os
15 from os import listdir
16 from PIL import Image as im
```

Figure 6.12 - Importing the libraries

The usual first step consists in importing the needed libraries:

a. TensorFlow is an open-source library to carry out deep learning tasks with Python [228].
b. TensorFlow Datasets is an API that provides a collection of ready-to-use datasets and several functions to work with datasets [229].
c. Keras is an open high-level Python API for neural networks, and it can run on top of TensorFlow library [230]. Importing the 'tensorflow.keras.preprocessing.image' module of Keras allows to perform preprocessing tasks over image datasets.
d. By importing the 'layers' module of Keras, it is possible to work with layers which are the basic building blocks of neural networks [231].

e. Scikit-learn is another open-source library providing tools for predictive data analysis. In particular, its 'sklearn.model_selection' module provides different methodologies to split the dataset into training and validation sets and perform an evaluation of the model, among which the Stratified K-Fold cross-validation technique is imported. The sklearn.metrics module of Scikit-learn library, instead, implements different functions to evaluate the performance of models and in particular, confusion_matrix scoring method, accuracy, and loss are imported to evaluate the accuracy of the classification [232].
f. NumPy is a library which allows to work with arrays, typically used as inputs to feed ML models [233].
g. Matplotlib library allows to create data visualizations and its 'pyplot' module, in particular, is able to graphically plot data [234].
h. As for the image concatenation code, Python built-in 'os' module is imported, which allows to interact with operating systems and the local directories on the device. In particular, the *os.listdir()* method is imported, which returns a list containing the names of the entries in a specific directory [225].
c. Pillow (PIL, Python Imaging Library) is a library providing image processing capabilities and, in particular, the 'PIL.Image' module which supplies the PIL image class and many correlated functions to manage images [226].

```
1  tf.keras.utils.set_random_seed(0)
```

Figure 6.13 - Seeding the random number generator

Generally, neural networks algorithms work on a stochastic basis, meaning that training the same network multiple times with the same dataset might yield different classification results (i.e., accuracy of predictions over the same validation and test sets can be different) [235]. In this way, accuracy results obtained from two different training procedures would not be comparable one with the other and it wouldn't be possible to understand which model setting is the most appropriate, since obtained accuracies would not be comparable and a better result could be simply due to randomness in the network (e.g., synapses weight are randomly initialized at the beginning of the training process).
This code line allows to seed the random number generator, making the program fully deterministic and reproducible so that the same results can be obtained every time the same network (with same hyperparameters setting) is applied to the same dataset (e.g., synapses weight are randomly initialized in the same way every time the same training process is started over) [236].

```
dataset = tf.keras.utils.image_dataset_from_directory('dataset_CV', labels='inferred', label_mode='binary',
                                                      class_names=["NO","SI"], color_mode='rgb', batch_size=32,
                                                      image_size=(64, 64))

count = 0

for element in dataset.as_numpy_iterator():
    if(count==0):
        x=element[0]
        y=element[1]
    else:
        x=np.append(x, element[0], axis=0)
        y=np.append(y, element[1], axis=0)
    count+=1
```

```
Found 70 files belonging to 2 classes.
```

Figure 6.14 - Importing the dataset

After these preliminary steps, the images dataset has been imported on Jupiter Notebook from a local folder and the binary labels (i.e., 'expulsion_YES' and 'expulsion_NO') are automatically inferred by the algorithm thanks to the two pre-labeled folders. Images size has been chosen on the basis of a trial-and-error approach: a larger size for the images made the algorithm strongly inefficient and running very slow, so to make it difficult to perform hyperparameters tuning and evaluate its performance and accuracy.
In this algorithm, the features within the images represent the independent variables since they are the known features of the problem under analysis, and the labels represent the dependent variables, as they are the targets that the algorithm needs to predict by classifying the images. The imported dataset contains both the images themselves and the automatically inferred labels: (image, label). In line with the format required by the CNN as input, two NumPy arrays are created, one containing the images and the other the labels (x and y respectively).

```
skf = StratifiedKFold(n_splits=5, random_state=0, shuffle=True)

cnn = tf.keras.models.Sequential()

cnn.add(tf.keras.layers.Rescaling(scale=1./255, offset=0.0))
cnn.add(tf.keras.layers.RandomFlip(mode="horizontal"))
cnn.add(tf.keras.layers.RandomZoom(height_factor=0.2, width_factor=0.2, fill_mode="reflect",
                                   interpolation="bilinear", fill_value=0.0))
cnn.add(tf.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu', input_shape=[64, 64, 3]))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
cnn.add(tf.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
cnn.add(tf.keras.layers.Flatten())
cnn.add(tf.keras.layers.Dense(units=8, activation='relu'))
#cnn.add(tf.keras.layers.Dense(units=8, activation='relu'))
cnn.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

optimizer = keras.optimizers.Adam(lr=0.001)
cnn.compile(optimizer = optimizer, loss = 'binary_crossentropy', metrics = ['accuracy'])
```

Figure 6.15 - Building the CNN model

Here, as a first step, the Stratified K-fold Cross-Validator is defined. The reason why Stratified K-fold cross-validation is used rather than plain K-fold cross-validation is that the former is an extension of the latter to be specifically used for classification problems.
With Stratified K-fold cross-validation, splits do not occur randomly, but rather the ratio between the target classes (i.e., 'expulsion_YES' vs 'expulsion_NO') is preserved and kept in each fold equal to the one in the complete dataset.
In principle, if the available dataset was large enough, regular K-fold cross-validation could be used since the ratio between the two classes would be more likely to be maintained within each fold, despite random splitting [237]. In the presented case, though, the dataset is not large enough for this reasoning to be valid and Stratified K-fold cross-validation seemed to be more appropriate.

Then, the Convolutional Neural Network architecture is finally built, by constructing a Sequential model via Keras:

a. A variable corresponding to the model itself is created and the CNN is initialized as a sequence of layers by defining it as an instance of a Keras Sequential object: an input layer, one hidden layer, and an output layer [238].
b. Feature scaling is performed by introducing a Keras rescaling layer, so to rescale all input images down to the same scale [239].

c. Two Keras Layers for data augmentation are included within the network: RandomFlip [240] and RandomZoom [241] layers to respectively flip (horizontally in this case) and zoom the input images so to augment the training set.
d. Two convolutional layers [242] and two max pooling layers [243] are added to the network, to respectively carry out the convolution and max pooling operations over the images. After having tried with higher amounts of filters and kernel sizes, each convolutional layer was finally set to contain 64 filters of 3x3 size which resulted in a good trade-off between accurate features extraction and computational efficiency. The pool size corresponds to the size of the matrix that is used in the Max Pooling operation, and in this case a 2x2 filter is applied. According to the specified 'strides' parameter, this window slides over the image by 2 pixels per time.
   After each convolution is carried out, ReLu activation function is applied to the resulting feature maps to break the linearity introduced by the convolution operation.
e. The flattening operation is carried out through a Keras Flatten Layer [244], to prepare the extracted features before being fed into the fully-connected portion of the CNN.
f. Then, 1 hidden layer is created as a Dense layer [245] with 8 units, while the input layer is implicitly created and will contain as many input neurons as the features of the input data. The choice of the number of hidden layers and hidden neurons will be discussed in the following section dedicated to hyperparameters tuning.
   The Dense objects are a specific typology of layers defined as fully connected layers, since they connect each node in the layer to each node belonging to the previous and to the following layers.
   As it is common practice, both hidden layers apply the ReLu activation function on the pre-activation output of their units, because of its implementation simplicity and effectiveness.
g. The output layer is built as a Dense layer as well, containing 1 single neuron since the algorithm is supposed to perform a binary classification of the input images into the two categories 'expulsion_YES' or 'expulsion_NO' and so, to predict a binary variable for which only 1 neuron is sufficient (taking value corresponding to 1 or 0 to distinguish the two classes). In line with this reasoning, Sigmoid function was chosen for this binary classification problem over Softmax activation function because of it higher computational simplicity: indeed, Sigmoid function allows to have one single output node with the probability of the input to belong to class 1, while SoftMax would have implied 2 output nodes, one corresponding to the probability of the output to belong to class 1 and the other to the probability of it to belong to class 0 [246].
h. After the model is built, it needs to be compiled by firstly specifying the 'optimizer' to be used: the best optimizers are the ones performing SDG, like the 'adam' optimizer that is a variant of the SDG optimization approach and is applied in this case study. The loss (or cost) function is specified, that is the way the difference between predictions and real values is computed during the training process: in this case, a 'binary_crossentropy' cost function was introduced, that is the one commonly introduced in case the algorithm is supposed to carry out a binary classification. Finally, evaluation metrics are chosen to evaluate the performance of the CNN, and in this specific algorithm the accuracy metric has been selected. Over the epochs, the accuracy evaluation metric is expected to incrementally grow while the loss function value is supposed to decrease.

```
accuracy_per_fold = []

accuracy_per_fold2 = []

epochs = 1000

for i, (train_index,validation_index) in enumerate(skf.split(x,y)):

    print(f'\nTraining for fold {i} ...')
    print(f'\nTraining set = {train_index}\n')
    print(y[train_index])
    print(f'\nValidation set = {validation_index}\n')
    print(y[validation_index])
    print('\n')

    history = cnn.fit(x[train_index], y[train_index], epochs=epochs, shuffle=True)

    #First approach: model.predict()

    scores = []
    y_predicted_probabilities = []

    y_predicted_probabilities = cnn.predict(x[validation_index])

    print(f'\nPredicted probabilities on fold {i}:\n')
    print(y_predicted_probabilities)

    y_predicted = []

    count = 0
    right_predictions = 0
    total_predictions = 14
    for element in y_predicted_probabilities:
        if element>=0.5:
            y_predicted.append(1)
        else:
            y_predicted.append(0)

        if y_predicted[count]==int((y[validation_index])[count]):
            right_predictions+=1
        count+=1

    accuracy = round(right_predictions/total_predictions,4)*100
    accuracy_per_fold.append(accuracy)

    print(f'\nPredicted labels on fold {i}:\n')
    print(y_predicted)
    print(f'\nTrue labels on fold {i}:\n')
    print(y[validation_index])

    print('\nAccuracy for fold {0}: {1}%'.format(i, accuracy))

    print('\nConfusion matrix:')
    cm=confusion_matrix(y[validation_index], y_predicted)
    print(cm)

    #Second approach: model.evaluate()

    scores = cnn.evaluate(x[validation_index], y[validation_index])
    print(f'\nAccuracy for fold {i}: {cnn.metrics_names[1]} of {round(scores[1]*100,2)}%')
    accuracy_per_fold2.append(scores[1]*100)

```

Figure 6.16 - Training the model over the training set

Two pair of empty lists are created, since they are going to contain the accuracy per each fold, computed with 2 different approaches, as it will be explained in the following.

At this point, Stratified K-fold cross-validation can start by iteratively splitting the dataset between a training set and a validation set, interchanging at each repetition the images included within these two sets. This is done within a 'for' loop in which the *i* index indicates the current K-fold cross-validation iteration, and the *StratifiedKFold.split()* method yields two sets: one containing the indices of the training set and the other the indices of the validation set resulting from the split.
At each iteration, the model is trained on the selected training set by calling the Keras *Model.fit* function. It firstly receives the NumPy arrays containing training set images and corresponding labels. Then, epochs are specified representing how many times the training set is passed through the model during the training process of each fold: in this case, for each of the 5 steps of K-Fold Cross-Validation, the training set is passed to the network 1000 times

(epochs). The choice of this number of epochs will be discussed in the following section dedicated to hyperparameters tuning. Finally, by specifying the 'shuffle' argument equal to True, the training data will be shuffled in a different order before each epoch: this is useful to avoid that the model learns any sequence from the data which might finally lead to overfitting.
As the model is under training, the progress of the training process over epochs and folds is presented as follows: the indexes of training set and validation set of the specific K-fold cross-validation iteration are displayed, and then, for each epoch, accuracy and loss of predictions over the training set are shown.

```
Training for fold 1 ...

Training set = [ 0  1  2  3  4  6  7  9 11 12 13 14 15 17 18 19 20 21 22 23 24 26 27 28
 30 33 34 35 38 40 42 43 44 45 46 47 48 49 50 52 53 54 55 56 57 58 59 60
 62 63 64 65 66 67 68 69]
                  Validation set = [ 5  8 10 16 25 29 31 32 36 37 39 41 51 61]
Epoch 1/1000
2/2 [==============================] - 0s 129ms/step - loss: 0.0740 - accuracy: 0.9643
Epoch 2/1000
2/2 [==============================] - 0s 137ms/step - loss: 0.1164 - accuracy: 0.9643
Epoch 3/1000
2/2 [==============================] - 0s 128ms/step - loss: 0.1860 - accuracy: 0.9464
Epoch 4/1000
2/2 [==============================] - 0s 124ms/step - loss: 0.0754 - accuracy: 0.9464
Epoch 5/1000
2/2 [==============================] - 0s 122ms/step - loss: 0.1199 - accuracy: 0.9821
Epoch 6/1000
2/2 [==============================] - 0s 124ms/step - loss: 0.1483 - accuracy: 0.9464
Epoch 7/1000
2/2 [==============================] - 0s 125ms/step - loss: 0.3319 - accuracy: 0.8750
Epoch 8/1000
2/2 [==============================] - 0s 134ms/step - loss: 0.2300 - accuracy: 0.8929
```

Figure 6.17 - Example of training process visualization

The next steps will be focused on the evaluation of the model over the validation set, with 2 different approaches. This has been done to have a double check over the procedure followed and indeed, in the end, both procedures yielded the same exact results in terms of accuracy of the model predictions across all the model configurations tested.

FIRST APPROACH: *model.predict()*

The *model.predict()* method of Keras framework returns the output predictions related to the input samples it receives as parameters: in this case, the method is fed with the validation set over which predictions need to be carried out so to evaluate the model performance.
Since the function yields the predictions in the form of probabilities of each sample in the validation set to belong to class 1, these probabilities are better be transformed into Boolean predictions, that is, equal to 1 if the sample image belongs to 'expulsion_YES' class and equal to 0 if it belongs to 'expulsion_NO' class. This is done thanks to a 'for' loop, which is also used to compare each prediction with the ground truth, that is, with the true label of each sample image (available since the dataset is labeled), so to count the right predictions performed.
After that, accuracy for this fold is derived as the fraction of right predictions over the total predictions performed: at each iteration, 14 predictions are performed on the validation set, because it is composed of 14 elements resulting from the split of the overall 70 samples into 5 folds of 14 samples (4 folds for the training, 1 for the validation). Accuracy is appended to

the initially created list of accuracies, containing accuracies derived through *model.predict()* method performed for each of the folds.
Then, the confusion matrix is derived for the specific fold, for a better understanding of correct and incorrect predictions over the validation fold.

```
Accuracy for fold 0: 78.57%

Confusion matrix:
[[2 3]
 [0 9]]
```

Figure 6.18 - Example of returned accuracy and confusion matrix by the algorithm

SECOND APPROACH: *model.evaluate()*

The *model.evaluate()* method keeps its computation more enclosed within itself with respect to *model.predict()*. Indeed, it simply receives features and labels of the validation set as parameters and yields the metrics corresponding to the predictions it internally performed. Again, accuracy is appended to a list containing all the accuracies derived through the *model.evaluate()* method for each of the folds.

By comparing the accuracies derived from *model.predict()* and *model.evaluate()*, it is possible to verify that they yield the same exact results, giving in this way more consistency to the followed procedure.

```
#First approach: model.predict()

print('Score per fold')
for i in range(0, len(accuracy_per_fold)):
    print(f'> Fold {i+1} - Accuracy: {round(accuracy_per_fold[i],2)}%')

print('\nAverage scores for all folds:')
print(f'> Accuracy: {round(np.mean(accuracy_per_fold),2)}% (+- {round(np.std(accuracy_per_fold),2)})')

#Second approach: model.evaluate()

print('Score per fold')
for i in range(0, len(accuracy_per_fold2)):
    print(f'> Fold {i+1} - Accuracy: {round(accuracy_per_fold2[i],2)}%')

print('\nAverage scores for all folds:')
print(f'> Accuracy: {round(np.mean(accuracy_per_fold2),2)}% (+- {round(np.std(accuracy_per_fold2),2)})')
```

Figure 6.19 - Average accuracies for the two validation methodologies

Starting from the lists containing accuracies per fold, the average accuracy is calculated for both the validation methods introduced and it is proven to be the same.

```
First approach: model.predict()

Score per fold
> Fold 1 - Accuracy: 78.57%
> Fold 2 - Accuracy: 100.0%
> Fold 3 - Accuracy: 100.0%
> Fold 4 - Accuracy: 85.71%
> Fold 5 - Accuracy: 100.0%

Average scores for all folds:
> Accuracy: 92.86% (+- 9.04)

Second approach: model.evaluate()

Score per fold
> Fold 1 - Accuracy: 78.57%
> Fold 2 - Accuracy: 100.0%
> Fold 3 - Accuracy: 100.0%
> Fold 4 - Accuracy: 85.71%
> Fold 5 - Accuracy: 100.0%

Average scores for all folds:
> Accuracy: 92.86% (+- 9.04)
```

Figure 6.20 - Accuracy resulted through the 2 validation methods for a specific hyperparameters configuration

```
correct_predictions = 0
total_predictions = 8

folder_dir = "test_set"

#fit su 70 immagini
cnn.fit(x, y, epochs=epochs, shuffle=True)

for im_name in os.listdir(folder_dir):

    if im_name != ".DS_Store":

        test_image = image.load_img('test_set/{0}'.format(im_name), target_size = (64, 64))
        test_image = image.img_to_array(test_image)
        test_image = np.expand_dims(test_image, axis = 0)
        result = cnn.predict(test_image)

        if result[0][0] == 1:
          prediction = 'SI'
        else:
          prediction = 'NO'

        print('\n')
        print(im_name)
        print(prediction)

        if ("SI" in im_name and prediction == "SI") or ("NO" in im_name and prediction == "NO"):
            print("Correct prediction\n")
            correct_predictions+=1
            #print(correct_predictions)
        else:
            print("Uncorrect prediction\n")

test_accuracy = round(correct_predictions/total_predictions,4)
print("\nTest accuracy: "+str(test_accuracy*100)+"%")
```

Figure 6.21 - Model run over the test set

After the K-fold cross-validation procedure is terminated, and the best model configuration has been derived accordingly through hyperparameters tuning (as described in the following section), the model is set to the chosen hyperparameters and trained from scratch over the whole dataset (except for the test set), without the need for distinguishing anymore between training and validation sets. Indeed, validation set is not useful anymore: it has been used for deriving the best model configuration through hyperparameters tuning; now that the best model setting has been found, such model needs to be trained; this training can be performed

over the whole initial dataset by keeping apart some datapoints for the final testing. As this training process comes to an end, the model is then ready to be applied to newly unseen and unlabeled data. The *model.predict()* function is called on the model to perform predictions over the test set which was set aside at the beginning. Within a 'for' loop, the algorithm understands whether the prediction made over a sample in the test set is correct or not by comparing it with the name with which the image was locally saved on the device (e.g., 52_NO) which contains an indication of the image label (without actually passing the algorithm the label information as it occurred for training and validation sets). In this way, a test accuracy can be computed as the ratio of correct predictions over total predictions performed on the test set samples.

```
20_NO.JPG
NO
Correct prediction

1/1 [==============================] - 0s 22ms/step


39_SI.JPG
SI
Correct prediction

1/1 [==============================] - 0s 25ms/step

Test accuracy: 87.5%
```

Figure 6.22 - Example of how performed predictions over the test set are displayed

6.8 Hyperparameters tuning

As anticipated, hyperparameters tuning is the process of running the network several times by changing its hyperparameters until the best setting is achieved in terms of prediction accuracy. This process allows to derive the best model configuration for the problem under analysis: depending on the chosen hyperparameters, and so, on the chosen model configuration, the algorithm will derive the network's internal parameters during the training process and achieve certain values for prediction accuracy accordingly.
So, it is clear how this tuning process is crucial for achieving the best accuracies possible, since it indirectly influences (by affecting the internal learnable parameters, like synapses weights) the outcome of the model in terms of final validation and test accuracies.

In this application, 3 hyperparameters have been chosen to perform the tuning, while all the others are kept constant throughout the whole process:

- Number of hidden layers (1 or 2)
- Number of neurons within the hidden layers (4, 8, or 12)
- Number of epochs (250, 500, 750, or 1000)

In order to have a clearer understanding of the behavior of the developed algorithm, the hyperparameters tuning and testing procedures have been repeated 5 times by interchanging images within the training, validation, and test sets: so, for each of the 5 procedures, the 3 datasets contained different images and, consequently, yielded different prediction accuracy results over both validation and test sets.

Each test set used was composed of 8 images, with same proportion between pictures showing expulsion (5) and those not showing any (3), so to keep consistency with the proportion between classes existing in the overall original dataset.

6.9 Analysis of the results

For each of the 5 testing procedures described above, validation accuracy and test accuracy have been calculated to derive the best model configuration, as they are directly influenced by the chosen model architecture and hyperparameters, and the quality, other than size, of the training set. Then, an analysis of the correct or incorrect prediction over each single image within each test set has been performed, so to study and understand that some images are more likely to be incorrectly predicted by the CNN than others, since harder to classify for some reason.

As an example, in the following a table is reported showing the resulting accuracies for different model configurations when performing the testing with the 'best' and the 'worst' selected test sets (in terms of fraction of images correctly classified over the different attempted model configurations).

						Test set 4							
	Hyperparameters			Validation		Test predictions							
Configuration	#hidden layers	#hidden neurons	#epochs	Validation accuracy	Test accuracy	23_SI	40_SI	62_SI	49_NO	19_NO	2_NO	69_SI	9_SI
1	1	4	250	87,14%	87,50%	No	Si	Si	No	No	No	Si	Si
2	1	4	500	92,86%	100,00%	Si	Si	Si	No	No	No	Si	Si
3	1	4	750	90,00%	100,00%	Si	Si	Si	No	No	No	Si	Si
4	1	4	1000	92,86%	87,50%	No	Si	Si	No	No	No	Si	Si
5	1	8	250	84,29%	75,00%	No	Si	No	No	No	No	Si	Si
6	1	8	500	88,57%	87,50%	Si	Si	Si	No	No	No	No	Si
7	1	8	750	94,29%	100,00%	Si	Si	Si	No	No	No	Si	Si
8	1	8	1000	88,57%	100,00%	Si	Si	Si	No	No	No	Si	SI
9	1	12	250	88,57%	62,50%	No	Si	No	No	No	No	No	Si
10	1	12	500	90,00%	87,50%	Si	Si	Si	No	No	No	No	Si
11	1	12	750	94,29%	100,00%	Si	Si	Si	No	No	No	Si	Si
12	1	12	1000	95,71%	100,00%	Si	Si	Si	No	No	No	Si	Si
13	2	4	250	82,86%	37,50%	No	No	No	No	No	No	No	No
14	2	4	500	84,29%	62,50%	No	Si	No	No	No	No	Si	No
15	2	4	750	90,00%	87,50%	No	Si	Si	No	No	No	Si	Si
16	2	4	1000	85,71%	87,50%	No	Si	Si	No	No	No	Si	Si
17	2	8	250	88,57%	50,00%	No	Si	No	No	No	No	No	No
18	2	8	500	95,71%	87,50%	No	Si	Si	No	No	No	Si	Si
19	2	8	750	95,71%	100,00%	Si	Si	Si	No	No	No	Si	Si
20	2	8	1000	91,43%	100,00%	Si	Si	Si	No	No	No	Si	Si
21	2	12	250	87,14%	50,00%	No	Si	No	No	No	No	No	No
22	2	12	500	87,14%	100,00%	Si	Si	Si	No	No	No	Si	Si
23	2	12	750	92,86%	100,00%	Si	Si	Si	No	No	No	Si	Si
24	2	12	1000	92,86%	100,00%	Si	Si	Si	No	No	No	Si	Si
					Correct predictions	13	23	18	24	24	24	18	20
					Total predictions	24	24	24	24	24	24	24	24
					Accuracy per image	54%	96%	75%	100%	100%	100%	75%	83%

Figure 6.23 - 'Best' test set

					Test set 2								
	Hyperparameters			Validation	Test predictions								
Attempt	#hidden layers	#hidden neurons	#epochs	Validation accuracy	Test accuracy	74_SI	47_NO	37_SI	45_SI	61_SI	54_NO	10_SI	8_NO
1	1	4	250	87,14%	37,50%	No	No	No	No	No	No	No	No
2	1	4	500	95,71%	75,00%	Si	No	Si	No	Si	No	No	No
3	1	4	750	98,57%	75,00%	Si	No	Si	No	Si	No	No	No
4	1	4	1000	94,29%	75,00%	Si	No	Si	No	Si	No	No	No
5	1	8	250	91,43%	62,50%	Si	No	No	No	Si	No	No	No
6	1	8	500	97,14%	75,00%	Si	No	Si	No	Si	No	No	No
7	1	8	750	94,29%	75,00%	Si	No	Si	No	Si	No	No	No
8	1	8	1000	97,14%	75,00%	Si	No	No	Si	Si	No	No	No
9	1	12	250	88,57%	62,50%	Si	No	No	No	Si	No	No	No
10	1	12	500	97,14%	75,00%	Si	No	Si	No	Si	No	No	No
11	1	12	750	94,29%	62,50%	Si	No	No	No	Si	No	No	No
12	1	12	1000	94,29%	62,50%	Si	No	No	No	Si	No	No	No
13	2	4	250	85,71%	37,50%	No	No	No	No	No	No	No	No
14	2	4	500	90,00%	37,50%	No	No	No	No	No	No	No	No
15	2	4	750	94,29%	75,00%	Si	No	Si	No	Si	No	No	No
16	2	4	1000	97,14%	37,50%	No	No	No	No	No	No	No	No
17	2	8	250	85,71%	50,00%	No	No	No	No	Si	No	No	No
18	2	8	500	90,00%	62,50%	Si	No	No	No	Si	No	No	No
19	2	8	750	90,00%	62,50%	Si	No	No	No	Si	No	No	No
20	2	8	1000	95,71%	75,00%	Si	No	Si	No	Si	No	No	No
21	2	12	250	85,71%	37,50%	No	No	No	No	No	No	No	No
22	2	12	500	87,14%	37,50%	No	No	No	No	No	No	No	No
23	2	12	750	90,00%	37,50%	No	No	No	No	No	No	No	No
24	2	12	1000	95,71%	50,00%	No	No	No	No	Si	No	No	No
					Correct predictions	15	24	8	1	17	24	0	24
					Total predictions	24	24	24	24	24	24	24	24
					Accuracy per image	63%	100%	33%	4%	71%	100%	0%	100%

Figure 6.24 – 'Worst' test set

The 5 tables used for this analysis (one per different test set introduced) are structured as follows:

- Identification number for the model configuration tested and hyperparameters chosen for that specific setting.
- Resulting validation accuracy derived through the K-fold cross-validation methodology.
- Resulting test accuracy computed as correct predictions over total predictions performed over the specific test set.
- Detail of how the individual images within the test set have been (correctly or incorrectly) classified by each model configuration, with computation of an accuracy per image defined as the fraction of times in which the specific image has been correctly predicted throughout all the model configurations attempted.

Then, the results of the 5 analyses have been displayed through graphs representing trends in validation accuracy and test accuracy as the number of epochs, number of hidden layers, and number of hidden neurons vary. To serve as an example, the graphs related to the Test set 3 case are reported and analyzed.

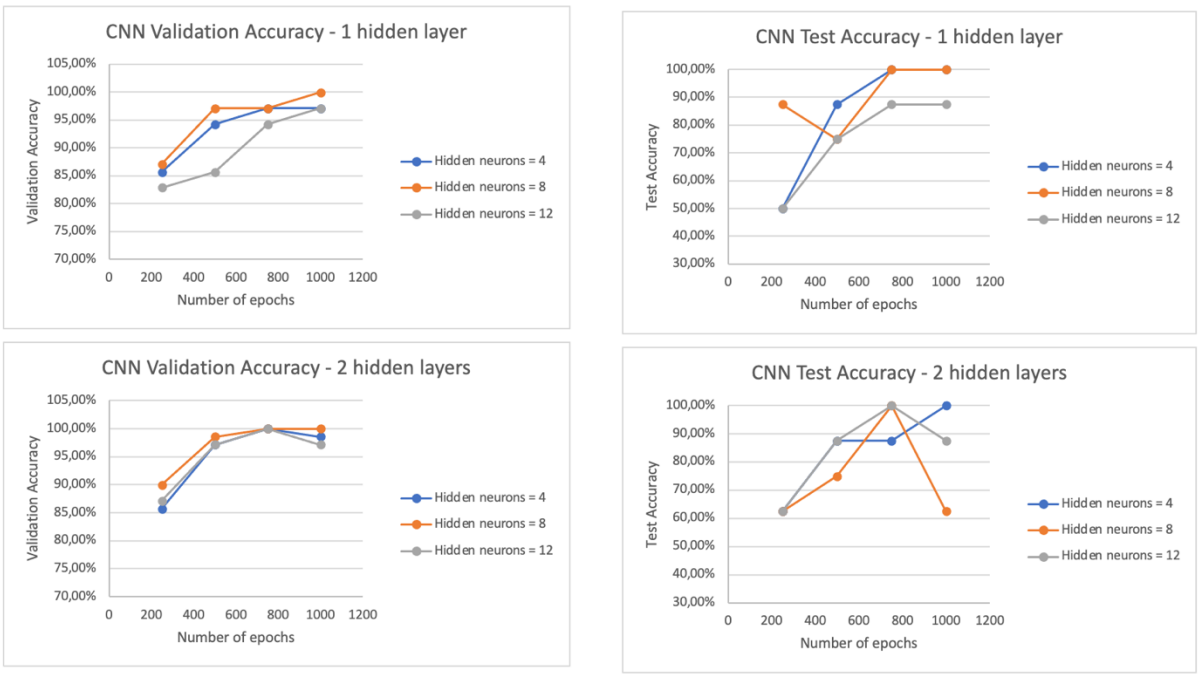

Figure 6.25 – Accuracy results from Test set 3

It can be noticed that variations in test set accuracy are wide from one configuration to the other: the reason is that being the test set so small (i.e., composed of 8 images), as soon as one algorithm configuration wrongly classifies 1 image more with respect to another setting, the test accuracy decreases by 12.5 percentage points.

In general, it can be noticed that when fixing the number of hidden layers and hidden neurons (together with all the other hyperparameters not specifically studied in this application) while varying the epochs, the validation and test accuracies increase with the number of epochs (or at least stay constant). This is due to the network refining its understanding of the training data and as such, making better predictions.
However, there are some exceptions to this general behavior.
A large number of epochs, indeed, does not always imply an increase in accuracy: increasing epochs might increase prediction accuracy up to a certain point, beyond which the model might start overfitting the training data and accuracy deteriorates [247] [248] [249].
Indeed, in those cases which show validation accuracy or test accuracy decreasing as the number of epochs increases, the network might be overfitting the training set: an epoch is a single pass of the complete training dataset through the network, which uses it to update its internal parameters; by increasing the number of epochs it might occur that the model gets too specialized to the specific training set by memorizing it and being uncapable to generalize what it learnt to new unseen datasets. The model gets too dependent on the features in the training set, by learning the noise within the data (i.e., patterns and features irrelevant for the classification of interest) and will generate higher prediction error rate over validation and test sets. For this reason, while the model keeps on performing better and better on the training set, its ability to perform correct classifications over validation and test sets gets worse, with validation and test accuracies staying constant or even decreasing (usually, test accuracy is lower than validation accuracy in case of overfitting).

So, the number of epochs is a crucial parameter to be set. If set too high, overfitting might arise; while if set too low, then underfitting might occur with the model not being trained for long enough to learn the patterns within the data and so, failing in capturing the features needed to perform satisfying predictions over all the datasets (training, validation, and test).

Another reason behind this exceptional behavior might be related to the relatively small training dataset available in this case study: it might be that such dataset is not large enough for the model to be trained and that the model is too complex compared to the training set size, so that the network will keep on learning from the same data without improving (or even worsening).

In principle, to choose the proper number of epochs for the model, attempts could be performed to see at what point overfitting starts arising, as it was done in this specific application through hyperparameters tuning via cross-validation. By experimenting different number of epochs and monitoring the resulting validation and test accuracies, the optimal number of epochs can be derived also depending on the specific model architecture chosen (e.g., number of hidden layers, number of hidden neurons).

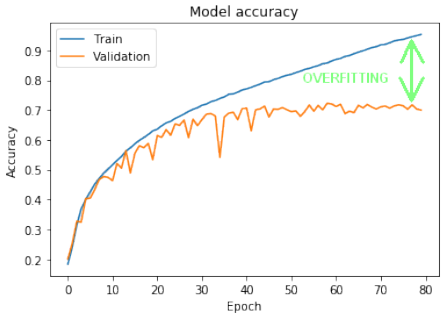

Figure 6.26 - Identification of overfitting [250]

However, as previously mentioned in the document, one possible way to address the choice of the right number of epochs and to avoid the overfitting issue is to introduce some regularization techniques like dropout or early stopping.
This can be easily done with classes provided by Keras (EarlyStopping [251]). EarlyStopping allows to interrupt the training process of the model when a specifically indicated metric has stopped improving: in case the validation loss is monitored, the network ends its training process as that metric starts increasing; in case the validation accuracy is monitored, the training process ends as that metric starts decreasing.

For what concerns the impact of number of hidden layers and number of hidden neurons on the model performance, in general, if the CNN has too few hidden layers, it might not be able to identify the complex patterns within the datasets so that it underfits the data, leading to poor validation and test accuracies. Too many hidden layers, instead, can make the model too complex so that it might start overfit the training data making validation and test accuracies deteriorate. So, the optimal number of hidden layers for the network depends on the specific problem and on the complexity of the dataset available: in the specific case, in which a CNN has been developed, the number of hidden layers affects the 'depth' and precision with which the network learns the features within the images. Typically, deeper CNNs can learn more

complex features, but it actually depends on the complexity of the problem to be solved: for simpler problems, like the one presented, also single-layer networks can perform good classifications. As it will be clearer in the last paragraphs of this chapter, 1 hidden layer seemed enough for this specific application, since by increasing the number of layers up to 2, accuracies didn't improve enough to justify the much higher computational cost and time introduced by the additional layer.

To understand the behavior of the model with respect to different input images, the 5 test sets have been picked of different 'difficulty' levels, proving that the model finds is difficult to identify the expulsion where it is not even clearly evident to human eyes.
Indeed, there are certain images where expulsion is barely visible that are never (or almost never) correctly classified by the algorithm: in line with the definition of expulsion, indeed, not necessarily the expulsion defect appears at the contact interface between sheet and electrode and is clearly visible; indeed, it might take place between the faying surfaces of the workpieces and be completely (or almost completely) hidden (in cases in which the ejection does not go outside the overlapped section of the workpieces).
Here some examples of those images that the algorithm finds difficult to classify no matter its model configuration (i.e., the chosen hyperparameters).

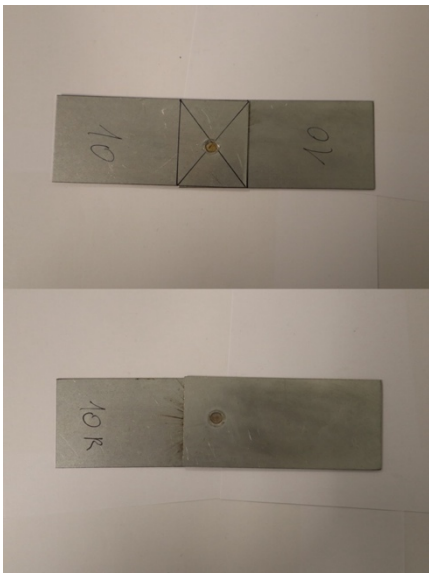

Figure 6.27 – Example of image difficult to be classified by the CNN

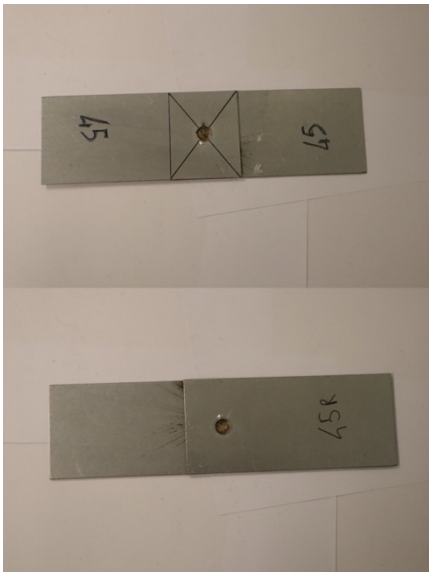

Figure 6.28 - Example of image difficult to be classified by the CNN

In conclusion, to identify the best model configuration for this problem, a summary table has been derived, showing the tried model configuration with average (validation and test) accuracies calculated over the 5 testing attempts performed.

	Hyperparameters			Average accuracies	
Configurations	#hidden layers	#hidden neurons	#epochs	Validation accuracy	Test accuracy
1	1	4	250	88,57%	55,00%
2	1	4	500	94,00%	82,50%
3	1	4	750	94,29%	82,50%
4	1	4	1000	94,57%	82,50%
5	1	8	250	88,29%	62,50%
6	1	8	500	93,43%	75,00%
7	1	8	750	94,57%	82,50%
8	1	8	1000	95,14%	85,00%
9	1	12	250	85,43%	52,50%
10	1	12	500	88,28%	77,50%
11	1	12	750	92,29%	82,50%
12	1	12	1000	94,57%	85,00%
13	2	4	250	87,14%	42,50%
14	2	4	500	90,57%	55,00%
15	2	4	750	93,43%	77,50%
16	2	4	1000	94,57%	72,50%
17	2	8	250	89,14%	50,00%
18	2	8	500	95,14%	70,00%
19	2	8	750	94,86%	85,00%
20	2	8	1000	95,71%	72,50%
21	2	12	250	89,14%	50,00%
22	2	12	500	91,43%	75,00%
23	2	12	750	94,29%	77,50%
24	2	12	1000	95,43%	77,50%

Figure 6.29 - Summary for model configuration choice

In line with these results, configurations 2, 3, 4, 7, 8, and 12 seem to be the best candidates to be chosen for the problem under analysis.
Indeed, despite some similarly good results can be achieved with configuration involving 2 hidden layers, choosing such configuration is not worth it in terms of computational costs and times. As an example, configuration 19 brings slightly better results than configuration 7:

however, given (i) the ‘simple’ task needed to be carried out to solve this problem (i.e., binary classification by identifying a relatively simple variable, that is the existence of expulsion or not) and (ii) the basically inexistent improvement in accuracy brought about by introducing 2 hidden layers rather than 1, it is more convenient to choose simpler configurations to reduce the network complexity (and consequently, computational costs and times) by still reaching comparably high accuracy values.
In a trade-off between relatively ‘low accuracy increase with higher computational costs and times’, and ‘not much lower accuracy with much lower computational costs and times’, the second scenario has been preferred when choosing the most suitable model configurations for the problem under analysis.

Conclusion

This thesis work led through a journey into the union between digital technologies and manufacturing processes, to discover how the approach in industrial operations is changing and evolving towards a more digitalized and automated setting.
Leading this transformation are Digital Twins with their ability of basically nullify the gap between real and virtual worlds, providing for a virtual mirror of reality in a way to help in the real environment management.
A theoretical overview of the concepts and a literature review made it possible to realize how the digital twin technology is typically applied in the field of welding processes: real-time monitoring, anomalies detection, real-time support in decision making, and many more. From this research process, it became clear how the integration of Machine Learning procedures within a Digital Twin framework is beneficial since making the framework itself 'Intelligent' and able to perform data-driven predictions and analyses.
To this end, this thesis work wishes to provide a practical exploration of the intersection between digital twin and machine learning (particularly deep learning) capabilities: a Convolutional Neural Network is introduced, tailored for image classification in Resistance Spot Welding and designed for detecting welding expulsion events. Such a model is not thought as an isolated solution, but rather as a building block which could enrich a broader digital twin framework for welding real-time quality monitoring, since expulsion can critically affect the final quality of the assemblies and should be kept under control as a quality index. It is not easy to recognize the goodness of a welding joint, and a digital representation of the welding process able to predict the quality of such joint could be strongly beneficial.
As revealed by the analysis of the network results, the classification accuracies are pretty good if one considers the relatively small size of the input images dataset with respect to the huge ones emerged from the literature review.

However, also limitations and challenges implied by the developed work should be acknowledged and future works should address such limitations by refining even more the CNN model accuracy and optimize its performance.
Firstly, the small size of the input dataset of images is clearly problematic and affecting the final accuracy values by the network. Future works should focus on the extraction of a much larger dataset, containing a larger number of diversified data points for the algorithm to train over a larger set of features and learn additional information to perform a more accurate classification.
Secondly, in the work presented a simple and straight-forward independent variable is derived from images features, that is the existence or absence of any expulsion trace on the final workpiece. Is such variable directly connected to the strength of the weld spot? Of course, the presence of expulsion could seriously impact the assembly strength, but not necessarily a workpiece classified as containing expulsion signs is not strong enough to continue to the following step of its manufacturing process. So, subsequent works should try to extract a more direct quality metric, able to directly assess the goodness of the welding joint, by even using the output variable extracted by the present model as input variable for another algorithm able to derive more direct quality indicators. Also, a more accurate algorithm could be able to predict a non-binary categorical variable not only to identify the simple presence of expulsion or not, but also its severity over a certain scale of values.

As a third point, evidently the process of input images extraction post-welding process should be automated as much as possible: in the present work, the focus was on understanding the functioning of a CNN, and for this purpose images were taken manually after the RSW process was ended. Understandably, this is not feasible over an industrial production line where the physical production process should not be delayed by the time needed by an operator to take pictures of the performed assemblies. The introduction of an industrial device for capturing existence or absence of expulsion during the welding process should be taken into account (e.g., an industrial camera for machine vision applications), so to accelerate the data extraction procedure.
Fourthly, some improvements can be obtained in terms of hyperparameters tuning. Indeed, much more hyperparameters could be tuned like the learning rate, the number of filters, and the size of kernels used for the convolution, and more. In addition, automated tuning approaches could be introduced within the algorithm, able to automatically derive the best model configuration in terms of hyperparameters, like Grid Search, Random Search, Bayesian optimization, etc.
Finally, the developed network is based on a supervised learning approach: this implies that an expert operator should manually classify the images within the training set by assigning 'expulsion' or 'no expulsion' labels to each image data point. Despite it would require a much larger input dataset, future works could focus on the development of a network based on an unsupervised learning approach to relieve the human operator of such preliminary time-consuming labeling duty.

In conclusion, this thesis work tries to give contribution but especially to highlight the challenges and opportunities ahead in a field which is wide and continuously evolving. A lot of research is still needed to understand the huge functionalities that digital twins and machine learning models can provide for improving monitoring and control over the manufacturing processes. The boundaries of these technologies should be pushed again and again, overcoming existing and future limitations, and embracing the limitless potential of Industry 4.0 enabling technologies in redesigning the future of the industrial landscape.

Bibliography

[1] 'Industry 4.0 Series: Manufacturing Then vs. Now'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.sangiacomo-presses.com/en/blog/industry-4.0-series-manufacturing-then-vs-now

[2] 'What is PLC ? Programmable Logic Controller - Unitronics'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.unitronicsplc.com/what-is-plc-programmable-logic-controller/

[3] 'AMCI : Advanced Micro Controls Inc :: What is a PLC?' Accessed: Sep. 28, 2023. [Online]. Available: https://www.amci.com/industrial-automation-resources/plc-automation-tutorials/what-plc/

[4] Q. Zhang, R. Xiao, Z. Liu, J. Duan, and J. Qin, 'Process Simulation and Optimization of Arc Welding Robot Workstation Based on Digital Twin', *Machines*, vol. 11, no. 1, 2023, doi: 10.3390/machines11010053.

[5] Q. Zhang, Z. Liu, J. Duan, and J. Qin, 'A Novel Method of Digital Twin-Based Manufacturing Process State Modeling and Incremental Anomaly Detection', *Machines*, vol. 11, no. 2, 2023, doi: 10.3390/machines11020151.

[6] 'What is Industry 4.0 and how does it work? | IBM'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/industry-4-0

[7] B. Marr, 'What is Industry 4.0? Here's A Super Easy Explanation For Anyone', Forbes. Accessed: Sep. 28, 2023. [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/

[8] 'What is Industry 4.0? Definition from SearchERP', ERP. Accessed: Sep. 28, 2023. [Online]. Available: https://www.techtarget.com/searcherp/definition/Industry-40

[9] 'What is industry 4.0 and the Fourth Industrial Revolution? | McKinsey'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-are-industry-4-0-the-fourth-industrial-revolution-and-4ir

[10] 'Industry 4.0: Digital transformation in manufacturing | McKinsey'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.mckinsey.com/capabilities/operations/our-insights/capturing-the-true-value-of-industry-four-point-zero

[11] 'Industrial Revolution - From Industry 1.0 to Industry 4.0'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.desouttertools.com/your-industry/news/503/industrial-revolution-from-industry-1-0-to-industry-4-0

[12] 'A Brief History of The 4 Industrial Revolutions | iED'. Accessed: Sep. 28, 2023. [Online]. Available: https://ied.eu/project-updates/the-4-industrial-revolutions/

[13] Q. Wang, W. Jiao, P. Wang, and Y. Zhang, 'Digital Twin for Human-Robot Interactive Welding and Welder Behavior Analysis', *IEEECAA J. Autom. Sin.*, vol. 8, no. 2, pp. 334–343, 2021, doi: 10.1109/JAS.2020.1003518.

[14] T. Ji and N. Mohamad Nor, 'Deep Learning-Empowered Digital Twin Using Acoustic Signal for Welding Quality Inspection', *Sensors*, vol. 23, no. 5, 2023, doi: 10.3390/s23052643.

[15] H. Li *et al.*, 'A detection and configuration method for welding completeness in the automotive body-in-white panel based on digital twin', *Sci. Rep.*, vol. 12, no. 1, 2022, doi: 10.1038/s41598-022-11440-0.

[16] 'Global investment in industrial digital transformation 2021 | Statista'. Accessed: Sep. 28, 2023. [Online]. Available: https://www-statista-com.ezproxy.biblio.polito.it/statistics/1358078/global-investment-in-industrial-digital-transformation/

[17] 'Framework for a Digital Twin in Manufacturing: Scope and Requirements - PMC'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7431924/#S1title

[18] 'Industry 4.0: The Future of Manufacturing', SAP. Accessed: Oct. 08, 2023. [Online]. Available: https://www.sap.com/products/scm/industry-4-0/what-is-industry-4-0.html

[19] 'What is the Internet of Things (IoT)?' Accessed: Sep. 28, 2023. [Online]. Available: https://www.oracle.com/internet-of-things/what-is-iot/

[20] 'Industrial Internet of Things: definizione, applicazioni e diffusione'. Accessed: Sep. 28, 2023. [Online]. Available: https://blog.osservatori.net/it_it/industrial-iot-definizione-applicazioni

[21] 'What is Industrial Cloud Computing? | Siemens Software'. Accessed: Sep. 28, 2023. [Online]. Available: https://www.plm.automation.siemens.com/global/en/our-story/glossary/industrial-cloud-computing/58773

[22] 'Industrial Cloud Computing: Scope and Future', HitechNectar. Accessed: Sep. 28, 2023. [Online]. Available: https://www.hitechnectar.com/blogs/industrial-cloud-computing-scope-and-future/

[23] 'Cloud computing', Salesforce. Accessed: Sep. 28, 2023. [Online]. Available: https://www.salesforce.com/it/learning-centre/tech/cloudcomputing/

[24] ‘What is cloud computing? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/cloud-computing
[25] ‘What Is Cloud Computing? | Microsoft Azure’. Accessed: Sep. 28, 2023. [Online]. Available: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-cloud-computing
[26] ‘Cos’è il cloud computing’. Accessed: Sep. 28, 2023. [Online]. Available: https://aws.amazon.com/it/what-is-cloud-computing/
[27] ‘What is edge computing? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/edge-computing
[28] ‘Edge Computing | Accenture’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.accenture.com/us-en/insights/cloud/edge-computing-index
[29] ‘What is Cybersecurity? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/cybersecurity
[30] ‘What Is Cybersecurity?’, Cisco. Accessed: Sep. 28, 2023. [Online]. Available: https://www.cisco.com/c/en/us/products/security/what-is-cybersecurity.html
[31] ‘Blockchain Meaning | Ledger’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ledger.com/academy/glossary/blockchain
[32] PricewaterhouseCoopers, ‘How can blockchain power industrial manufacturing?’, PwC. Accessed: Sep. 28, 2023. [Online]. Available: https://www.pwc.com/us/en/industries/industrial-products/library/blockchain-industrial-manufacturing.html
[33] ‘What is Blockchain Technology? - IBM Blockchain | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/blockchain
[34] ‘What is Advanced Analytics? | TIBCO Software’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.tibco.com/reference-center/what-is-advanced-analytics
[35] ‘Definition of Advanced Analytics - IT Glossary | Gartner’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.gartner.com/en/information-technology/glossary/advanced-analytics
[36] ‘What is Advanced Analytics in Manufacturing?’ Accessed: Sep. 28, 2023. [Online]. Available: https://www.precog.co/glossary/advanced-analytics/
[37] ‘Data Science vs Advanced Analytics | Know Everything Here’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.xenonstack.com/blog/data-science-vs-advanced
[38] ‘Data Science vs Data Analytics: Definitions and Differences’, Qlik. Accessed: Sep. 28, 2023. [Online]. Available: https://www.qlik.com/us/data-analytics/data-science-vs-data-analytics
[39] ‘Virtual reality (VR) | Definition, Development, Technology, Examples, & Facts | Britannica’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.britannica.com/technology/virtual-reality
[40] ‘Augmented Reality (AR) Defined, With Examples and Uses’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.investopedia.com/terms/a/augmented-reality.asp
[41] ‘What is augmented reality (AR)? | SAP’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.sap.com/products/scm/industry-4-0/what-is-augmented-reality.html
[42] ‘What’s the Difference Between AR and VR?’, Tulane School of Professional Advancement. Accessed: Sep. 28, 2023. [Online]. Available: https://sopa.tulane.edu/blog/whats-difference-between-ar-and-vr
[43] ‘Virtual Reality, the technology of the future - Iberdrola’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.iberdrola.com/innovation/virtual-reality
[44] ‘AR & VR Applications in Manufacturing: Redefining Productivity and Quality’. Accessed: Sep. 28, 2023. [Online]. Available: https://smarttek.solutions/blog/ar-and-vr-in-manufacturing-use-cases-and-benefits/
[45] ‘Robotics: What Are Robots? Robotics Definition & Uses. | Built In’. Accessed: Sep. 28, 2023. [Online]. Available: https://builtin.com/robotics
[46] ‘Robotics | Definition, Applications, & Facts | Britannica’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.britannica.com/technology/robotics
[47] ‘What is Robotics? - Definition from Techopedia’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.techopedia.com/definition/32836/robotics
[48] ‘What is automation? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/automation
[49] ‘What is Industrial Automation and Robotics?’ Accessed: Sep. 28, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-industrial-automation-and-robotics.aspx
[50] ‘What is Additive Manufacturing? (Definition & Types)’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-additive-manufacturing.aspx
[51] ‘Additive manufacturing, explained | MIT Sloan’. Accessed: Sep. 28, 2023. [Online]. Available: https://mitsloan.mit.edu/ideas-made-to-matter/additive-manufacturing-explained

[52] ‘What is Additive Manufacturing | GE Additive’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ge.com/additive/additive-manufacturing
[53] ‘AM Basics - Additive Manufacturing (AM)’. Accessed: Sep. 28, 2023. [Online]. Available: https://additivemanufacturing.com/basics/
[54] ‘What is Artificial Intelligence (AI) ? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/artificial-intelligence
[55] ‘Artificial Intelligence (AI): What Is AI and How Does It Work? | Built In’. Accessed: Sep. 28, 2023. [Online]. Available: https://builtin.com/artificial-intelligence
[56] ‘Top 18 Artificial Intelligence (AI) Applications in 2023 | Simplilearn’, Simplilearn.com. Accessed: Sep. 28, 2023. [Online]. Available: https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/artificial-intelligence-applications
[57] ‘In cosa consiste il Machine Learning? | Oracle Italia’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.oracle.com/it/artificial-intelligence/machine-learning/what-is-machine-learning/
[58] ‘What is a digital twin? | IBM’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.ibm.com/topics/what-is-a-digital-twin
[59] ‘What is Digital Twin Technology and How Does it Work? - TWI’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-digital-twin
[60] K. Moran, ‘Benefits of Industry 4.0’, SL Controls. Accessed: Sep. 28, 2023. [Online]. Available: https://slcontrols.com/benefits-of-industry-4-0/
[61] ‘Industry 4.0 in a Nutshell. | SAP Blogs’. Accessed: Sep. 28, 2023. [Online]. Available: https://blogs.sap.com/2020/05/31/industry-4.0-in-a-nut-shell./
[62] ‘Benefits of Industry 4.0 for Manufacturers | HighGear’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.highgear.com/blog/benefits-of-industry-4-0/
[63] ‘What is industry 4.0? | Definition, technologies, benefits’, SAP. Accessed: Sep. 28, 2023. [Online]. Available: https://www.sap.com/products/scm/industry-4-0/what-is-industry-4-0.html
[64] Gonzalo, ‘Industry 4.0 | Advantages and disadvantages’, BECOSAN®. Accessed: Sep. 28, 2023. [Online]. Available: https://www.becosan.com/industry-4-0-the-fourth-industrial-revolution/
[65] T. H. C. Hub, ‘How Digitalization Improves Workplace Safety’, Human Capital Hub. Accessed: Sep. 28, 2023. [Online]. Available: https://thehumancapitalhub.com/articles/how-digitalization-improves-workplace-safety
[66] ‘Industry 4.0 challenges and risks | nibusinessinfo.co.uk’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.nibusinessinfo.co.uk/content/industry-4.0-challenges-and-risks
[67] ‘Employment impact of digitalisation | European Foundation for the Improvement of Living and Working Conditions’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.eurofound.europa.eu/en/employment-impact-digitalisation
[68] ‘Industry 5.0’. Accessed: Sep. 28, 2023. [Online]. Available: https://research-and-innovation.ec.europa.eu/research-area/industrial-research-and-innovation/industry-50_en
[69] ‘Industry 5.0: Adding the human edge to industry 4.0 | SAP insights’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.sap.com/insights/industry-5-0.html
[70] ‘Industria 5.0: cos’è e come impatterà sulle aziende - CorCom’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.corrierecomunicazioni.it/digital-economy/industria-5-0-cose-e-come-impattera-sulle-aziende/
[71] ‘What is Industry 5.0? (Top 5 Things You Need To Know) - TWI’. Accessed: Sep. 28, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/industry-5-0
[72] E. VanDerHorn and S. Mahadevan, ‘Digital Twin: Generalization, characterization and implementation’, *Decis. Support Syst.*, vol. 145, p. 113524, Jun. 2021, doi: 10.1016/j.dss.2021.113524.
[73] G. Chen, J. Zhu, Y. Zhao, Y. Hao, C. Yang, and Q. Shi, ‘Digital twin modeling for temperature field during friction stir welding’, *J. Manuf. Process.*, vol. 64, pp. 898–906, 2021, doi: 10.1016/j.jmapro.2021.01.042.
[74] B. Maschler, D. Braun, N. Jazdi, and M. Weyrich, ‘Transfer learning as an enabler of the intelligent digital twin’, presented at the Procedia CIRP, 2021, pp. 127–132. doi: 10.1016/j.procir.2021.05.020.
[75] H. Li *et al.*, ‘Real-time detection method for welding parts completeness based on improved YOLOX in a digital twin environment’, *Meas. Sci. Technol.*, vol. 34, no. 5, p. 055004, May 2023, doi: 10.1088/1361-6501/acb0ee.
[76] D. Maity, R. Premchand, M. Muralidhar, and V. Racherla, ‘Real-time temperature monitoring of weld interface using a digital twin approach’, *Meas. J. Int. Meas. Confed.*, vol. 219, 2023, doi: 10.1016/j.measurement.2023.113278.

[77] H. Hultman, S. Cedergren, R. Söderberg, and K. Wärmefjord, 'Towards a digital twin setup for individualized production of fabricated components', presented at the ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE), 2021. doi: 10.1115/IMECE2021-70212.
[78] W. Hu, T. Zhang, X. Deng, Z. Liu, and J. Tan, 'Digital twin: a state-of-the-art review of its enabling technologies, applications and challenges', *J. Intell. Manuf. Spec. Equip.*, vol. 2, no. 1, pp. 1–34, Jan. 2021, Accessed: Sep. 29, 2023. [Online]. Available: https://doi.org/10.1108/JIMSE-12-2020-010
[79] G. Shao, S. P. Frechette, and V. Srinivasan, 'An Analysis of the New ISO 23247 Series of Standards on Digital Twin Framework for Manufacturing', *NIST*, Jun. 2023, Accessed: Sep. 29, 2023. [Online]. Available: https://www.nist.gov/publications/analysis-new-iso-23247-series-standards-digital-twin-framework-manufacturing
[80] 'Digital Twin - Global Market and Forecast Till 2030'. Accessed: Sep. 29, 2023. [Online]. Available: https://www.acumenresearchandconsulting.com/digital-twin-market
[81] J. Dong, J. Hu, and Z. Luo, 'Quality Monitoring of Resistance Spot Welding Based on a Digital Twin', *Metals*, vol. 13, no. 4, 2023, doi: 10.3390/met13040697.
[82] 'Advancing digital twin implementation: a toolbox for modelling and simulation - ScienceDirect'. Accessed: Sep. 29, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212827121003656
[83] 'Apollo 13: The First Digital Twin - Simcenter'. Accessed: Sep. 29, 2023. [Online]. Available: https://blogs.sw.siemens.com/simcenter/apollo-13-the-first-digital-twin/
[84] M. Grieves and J. Vickers, 'Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems', in *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, 2016, pp. 85–113. doi: 10.1007/978-3-319-38756-7_4.
[85] M. Shafto *et al.*, *Modeling, Simulation, Information Technology and Processing Roadmap*. 2010.
[86] A. Agrawal, R. Thiel, P. Jain, V. Singh, and M. Fischer, 'Digital Twin: Where do humans fit in?' arXiv, Jan. 08, 2023. doi: 10.48550/arXiv.2301.03040.
[87] 'Digital Twin: Origin to Future'. Accessed: Sep. 29, 2023. [Online]. Available: https://encyclopedia.pub/entry/11253
[88] F. Tao and M. Zhang, 'Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing', *IEEE Access*, vol. 5, pp. 20418–20427, 2017, doi: 10.1109/ACCESS.2017.2756069.
[89] 'Decoding Digital Twins: Exploring the 6 main applications and their benefits', IoT Analytics. Accessed: Sep. 30, 2023. [Online]. Available: https://iot-analytics.com/6-main-digital-twin-applications-and-their-benefits/
[90] 'What Is a Digital Twin? Definition, Types, and Uses | Coursera'. Accessed: Sep. 30, 2023. [Online]. Available: https://www.coursera.org/articles/digital-twin
[91] 'The types of Digital Twins in manufacturing - Braincube'. Accessed: Sep. 30, 2023. [Online]. Available: https://braincube.com/resource/the-types-of-digital-twins-in-manufacturing/
[92] 'ISO/DIS 23247-4(en), Automation systems and integration — Digital Twin framework for manufacturing — Part 4: Information exchange'. Accessed: Sep. 30, 2023. [Online]. Available: https://www.iso.org/obp/ui#iso:std:iso:23247:-4:dis:ed-1:v1:en
[93] 'What is the ISO (International Organization for Standardization)?' Accessed: Sep. 30, 2023. [Online]. Available: https://www.techtarget.com/searchdatacenter/definition/ISO
[94] 'Digitalization in industry: Twins with potential', siemens.com Global Website. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/global/en/company/stories/industry/the-digital-twin.html
[95] E. Newton, '7 Sectors Seeing Remarkable Results from Digital Twins', Revolutionized. Accessed: Sep. 30, 2023. [Online]. Available: https://revolutionized.com/industrial-sector-digital-twins/
[96] 'Digital Twin Market Size, Trends, Growth & Forecast [2030]'. Accessed: Sep. 30, 2023. [Online]. Available: https://www.fortunebusinessinsights.com/digital-twin-market-106246
[97] 'How Digital Twins are Transforming Aerospace & Defense | Synopsys Blog'. Accessed: Sep. 30, 2023. [Online]. Available: https://www.synopsys.com/blogs/chip-design/digital-twins-transform-aerospace-defense.html
[98] 'What is a digital patient twin?' Accessed: Sep. 30, 2023. [Online]. Available: https://www.siemens-healthineers.com/perspectives/digital-patient-twin
[99] 'Digital twin in Sanità: quale futuro?', Healthtech360. Accessed: Sep. 30, 2023. [Online]. Available: https://www.healthtech360.it/digital-twin/digital-twin-in-sanita/
[100] ecanorea, 'Why are Digital Twins essential for Smart Cities?', Plain Concepts. Accessed: Sep. 30, 2023. [Online]. Available: https://www.plainconcepts.com/digital-twins-smart-cities/

[101] ‘About Siemens’, siemens.com Global Website. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/global/en/company/about.html
[102] ‘Digital Industries’, Siemens Italia. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/it/it/azienda/chi-siamo/business/digital-industries.html
[103] ‘Siemens Digital Industries Software’, Siemens Digital Industries Software. Accessed: Oct. 01, 2023. [Online]. Available: https://www.sw.siemens.com/it-IT/
[104] ‘Simulation and Digital Twin’, siemens.com Global Website. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/global/en/company/innovation/siemens-core-technologies/simulation-digital-twin.html
[105] ‘Digital Twins everywhere’, siemens.com Global Website. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/global/en/company/stories/research-technologies/digitaltwin/outlook2030.html
[106] ‘Digital Enterprise’, Siemens Italia. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/it/it/azienda/temi-chiave/digital-enterprise.html
[107] ‘Digital twin and simulation in the pharma industry’, siemens.com Global Website. Accessed: Oct. 01, 2023. [Online]. Available: https://www.siemens.com/global/en/industries/pharmaceutical-life-science-industries/pharma-industry/focus-topics/digital-twin.html
[108] ‘Digital Twin | Siemens’, Siemens Digital Industries Software. Accessed: Oct. 01, 2023. [Online]. Available: https://www.plm.automation.siemens.com/global/it/our-story/glossary/digital-twin/24465
[109] ‘Il Digital Twin nella produzione’, Siemens Digital Industries Software. Accessed: Oct. 01, 2023. [Online]. Available: https://webinars.sw.siemens.com/it-IT/digital-twin-in-manufacturing/
[110] ‘Che cos’è e come funziona Xcelerator, la transizione digitale ed energetica secondo Siemens | Sky TG24’. Accessed: Oct. 01, 2023. [Online]. Available: https://tg24.sky.it/tecnologia/now/2023/05/04/siemens-xcelerator-transizione-digitale-imprese-come-funziona
[111] ‘Siemens Xcelerator, la tua strategia di trasformazione digitale | Siemens Software’. Accessed: Oct. 01, 2023. [Online]. Available: https://www.sw.siemens.com/it-IT/digital-transformation/
[112] ‘Maximo Application Suite | IBM’. Accessed: Oct. 01, 2023. [Online]. Available: https://www.ibm.com/it-it/products/maximo
[113] W. Beneke, ‘Microsoft Azure Digital Twins : Everything You Need To Know’, XMPRO. Accessed: Oct. 01, 2023. [Online]. Available: https://xmpro.com/microsoft-azure-digital-twins-everything-you-need-to-know/
[114] W. A. Hussain, ‘15 Best Digital Twin Companies To Look For In 2023’, The Metaverse Insider. Accessed: Oct. 01, 2023. [Online]. Available: https://metaverseinsider.tech/2023/03/08/best-digital-twin-companies/
[115] baanders, ‘What is Azure Digital Twins? - Azure Digital Twins’. Accessed: Oct. 01, 2023. [Online]. Available: https://learn.microsoft.com/en-us/azure/digital-twins/overview
[116] ‘(11) Global Digital Twin Market Size, Trends and Outlook for 2023-2030 | LinkedIn’. Accessed: Oct. 01, 2023. [Online]. Available: https://www.linkedin.com/pulse/global-digital-twin-market-size-trends-outlook-2023-2030/?trk=article-ssr-frontend-pulse_more-articles_related-content-card
[117] ‘Discover Top 8 Digital Twin Trends in 2023’. Accessed: Oct. 01, 2023. [Online]. Available: https://research.aimultiple.com/digital-twin-trends/
[118] ‘Gartner Survey Reveals Digital Twins Are Entering Mainstream Use’, Gartner. Accessed: Oct. 01, 2023. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2019-02-20-gartner-survey-reveals-digital-twins-are-entering-mai
[119] J. Eichberger, ‘Learnings From The Digital Twin’s Data Architecture Of Tesla’, Cloudflight. Accessed: Oct. 29, 2023. [Online]. Available: https://www.cloudflight.io/en/blog/learnings-from-the-digital-twins-data-architecture-of-tesla/
[120] ‘(11) Digital Twins and Cybersecurity: Risks and Mitigation in the Age of Industrial Digitization | LinkedIn’. Accessed: Oct. 01, 2023. [Online]. Available: https://www.linkedin.com/pulse/digital-twins-cybersecurity-risks-mitigation-age-industrial-hagen/
[121] ‘Machine Learning A-Z (Python & R in Data Science Course)’, Udemy. Accessed: Oct. 01, 2023. [Online]. Available: https://www.udemy.com/course/machinelearning/
[122] K. Nyuytiymbiy, ‘The Stages of a Machine Learning Project’, The Startup. Accessed: Oct. 01, 2023. [Online]. Available: https://medium.com/swlh/the-stages-of-a-machine-learning-project-cf4bb073a4ad

[123] ‘Why One-Hot Encode Data in Machine Learning? - MachineLearningMastery.com’. Accessed: Oct. 01, 2023. [Online]. Available: https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/
[124] ‘Test, training and validation sets’, BrainsToBytes. Accessed: Oct. 01, 2023. [Online]. Available: https://www.brainstobytes.com/test-training-and-validation-sets/
[125] K. Nyuytiymbiy, ‘Parameters and Hyperparameters in Machine Learning and Deep Learning’, Medium. Accessed: Oct. 01, 2023. [Online]. Available: https://towardsdatascience.com/parameters-and-hyperparameters-aa609601a9ac
[126] D. Castillo, ‘Machine Learning Regression Explained’, Seldon. Accessed: Oct. 02, 2023. [Online]. Available: https://www.seldon.io/machine-learning-regression-explained
[127] ‘Regression vs. Classification in Machine Learning: What’s the Difference?’, Springboard Blog. Accessed: Oct. 02, 2023. [Online]. Available: https://www.springboard.com/blog/data-science/regression-vs-classification/
[128] A. Raj, ‘Unlocking the True Power of Support Vector Regression’, Medium. Accessed: Oct. 02, 2023. [Online]. Available: https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0
[129] B. O. T. Ph.D, ‘Linear Regression Basics for Absolute Beginners’, Medium. Accessed: Oct. 09, 2023. [Online]. Available: https://pub.towardsai.net/linear-regression-basics-for-absolute-beginners-68ed9ff980ae
[130] K. Kargın, ‘Multiple Linear Regression Fundamentals and Modeling in Python’, MLearning.ai. Accessed: Oct. 09, 2023. [Online]. Available: https://medium.com/mlearning-ai/multiple-linear-regression-fundamentals-and-modeling-in-python-60db7095deff
[131] ‘Python Machine Learning Polynomial Regression’. Accessed: Nov. 13, 2023. [Online]. Available: https://www.w3schools.com/python/python_ml_polynomial_regression.asp
[132] ‘Support Vector Regression In Machine Learning’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/03/support-vector-regression-tutorial-for-machine-learning/
[133] H. L. and M. Li, *11.4 Regression and Decision Tree Basic | Practitioner’s Guide to Data Science*. Accessed: Oct. 09, 2023. [Online]. Available: http://scientistcafe.com/IDS/
[134] ‘Random Forest Regression. Random Forest Regression is a… | by Chaya | Level Up Coding’. Accessed: Oct. 09, 2023. [Online]. Available: https://levelup.gitconnected.com/random-forest-regression-209c0f354c84
[135] O. Harrison, ‘Machine Learning Basics with the K-Nearest Neighbors Algorithm’, Medium. Accessed: Oct. 02, 2023. [Online]. Available: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
[136] D. Subramanian, ‘A Simple Introduction to K-Nearest Neighbors Algorithm’, Medium. Accessed: Oct. 02, 2023. [Online]. Available: https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e
[137] *Logistic Regression: An Introduction*, (Feb. 05, 2021). Accessed: Oct. 02, 2023. [Online Video]. Available: https://www.youtube.com/watch?v=3tq4t41MsPc
[138] R. Gandhi, ‘Support Vector Machine — Introduction to Machine Learning Algorithms’, Medium. Accessed: Oct. 02, 2023. [Online]. Available: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
[139] ‘Major Kernel Functions in Support Vector Machine (SVM)’, GeeksforGeeks. Accessed: Oct. 02, 2023. [Online]. Available: https://www.geeksforgeeks.org/major-kernel-functions-in-support-vector-machine-svm/
[140] R. Gandhi, ‘Naive Bayes Classifier’, Medium. Accessed: Oct. 02, 2023. [Online]. Available: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
[141] ‘K-Nearest Neighbor(KNN) Algorithm - GeeksforGeeks’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.geeksforgeeks.org/k-nearest-neighbours/
[142] ‘Support Vector Machine (SVM) Algorithm - Javatpoint’. Accessed: Oct. 09, 2023. [Online]. Available: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm
[143] www.aionlinecourse.com, ‘Kernel SVM for Dummies(with Python Code)’, www.aionlinecourse.com. Accessed: Oct. 09, 2023. [Online]. Available: https://www.aionlinecourse.com/tutorial/machine-learning/kernel-svm
[144] M. Elzeiny, ‘The ultimate guide to Naive Bayes’, Machine Learning Archive. Accessed: Oct. 09, 2023. [Online]. Available: https://mlarchive.com/machine-learning/the-ultimate-guide-to-naive-bayes/

[145] M. Cavaioni, 'Machine Learning: Decision Tree Classifier', Machine Learning bites. Accessed: Oct. 09, 2023. [Online]. Available: https://medium.com/machine-learning-bites/machine-learning-decision-tree-classifier-9eb67cad263e
[146] S. Awasthi, 'Random Forests in Machine Learning: A Detailed Explanation', datamahadev.com. Accessed: Oct. 09, 2023. [Online]. Available: https://datamahadev.com/random-forests-in-machine-learning-a-detailed-explanation/
[147] 'R-Squared', Corporate Finance Institute. Accessed: Oct. 03, 2023. [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/r-squared/
[148] 'Residual Sum of Squares (RSS): What It Is, How to Calculate It', Investopedia. Accessed: Oct. 03, 2023. [Online]. Available: https://www.investopedia.com/terms/r/residual-sum-of-squares.asp
[149] A. Bhandari, 'Key Difference between R-squared and Adjusted R-squared for Regression Analysis', Analytics Vidhya. Accessed: Oct. 03, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/07/difference-between-r-squared-and-adjusted-r-squared/
[150] S. Narkhede, 'Understanding Confusion Matrix', Medium. Accessed: Oct. 03, 2023. [Online]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
[151] 'What is Overfitting? | IBM'. Accessed: Oct. 03, 2023. [Online]. Available: https://www.ibm.com/topics/overfitting
[152] 'What is Overfitting? - Overfitting in Machine Learning Explained - AWS', Amazon Web Services, Inc. Accessed: Oct. 03, 2023. [Online]. Available: https://aws.amazon.com/what-is/overfitting/
[153] '8 Simple Techniques to Prevent Overfitting | by David Chuan-En Lin | Towards Data Science'. Accessed: Oct. 03, 2023. [Online]. Available: https://towardsdatascience.com/8-simple-techniques-to-prevent-overfitting-4d443da2ef7d#c287
[154] 'Keras dropout layer: implement regularization', Educative. Accessed: Nov. 06, 2023. [Online]. Available: https://www.educative.io/answers/keras-dropout-layer-implement-regularization
[155] 'Dropout in (Deep) Machine learning | by Amar Budhiraja | Medium'. Accessed: Oct. 09, 2023. [Online]. Available: https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5
[156] 'Papers with Code - Early Stopping Explained'. Accessed: Oct. 09, 2023. [Online]. Available: https://paperswithcode.com/method/early-stopping
[157] EliteDataScience, 'Overfitting in Machine Learning: What It Is and How to Prevent It', EliteDataScience. Accessed: Nov. 13, 2023. [Online]. Available: https://elitedatascience.com/overfitting-in-machine-learning
[158] 'Cross Validation Explained: Evaluating estimator performance. | by Rahil Shaikh | Towards Data Science'. Accessed: Oct. 09, 2023. [Online]. Available: https://towardsdatascience.com/cross-validation-explained-evaluating-estimator-performance-e51e5430ff85
[159] 'What is Underfitting? | IBM'. Accessed: Oct. 03, 2023. [Online]. Available: https://www.ibm.com/topics/underfitting
[160] 'Underfitting', DataRobot AI Platform. Accessed: Oct. 03, 2023. [Online]. Available: https://www.datarobot.com/wiki/underfitting/
[161] I. D. and A. Team, 'AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the difference?', IBM Blog. Accessed: Oct. 03, 2023. [Online]. Available: https://www.ibm.com/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks/
[162] 'Deep learning vs. machine learning – what are the differences?', IONOS Digital Guide. Accessed: Oct. 03, 2023. [Online]. Available: https://www.ionos.com/digitalguide/online-marketing/search-engine-marketing/deep-learning-vs-machine-learning/
[163] 'Apprendimento profondo vs. machine learning: Qual è la differenza?', Zendesk IT. Accessed: Oct. 03, 2023. [Online]. Available: https://www.zendesk.com/it/blog/machine-learning-and-deep-learning/
[164] 'Supervised vs. Unsupervised Learning [Differences & Examples]'. Accessed: Oct. 03, 2023. [Online]. Available: https://www.v7labs.com/blog/supervised-vs-unsupervised-learning, https://www.v7labs.com/blog/supervised-vs-unsupervised-learning
[165] J. Delua, 'Supervised vs. Unsupervised Learning: What's the Difference?', IBM Blog. Accessed: Oct. 03, 2023. [Online]. Available: https://www.ibm.com/blog/supervised-vs-unsupervised-learning/
[166] 'Difference between Supervised vs Unsupervised Learning', Hackr.io. Accessed: Oct. 03, 2023. [Online]. Available: https://hackr.io/blog/supervised-vs-unsupervised-learning
[167] R. Bevans, 'Multiple Linear Regression | A Quick Guide (Examples)', Scribbr. Accessed: Oct. 31, 2023. [Online]. Available: https://www.scribbr.com/statistics/multiple-linear-regression/

[168] W. Badr, 'Auto-Encoder: What Is It? And What Is It Used For? (Part 1)', Medium. Accessed: Oct. 03, 2023. [Online]. Available: https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726
[169] 'Figure 1: Example for an artificial neural network with two input...', ResearchGate. Accessed: Oct. 03, 2023. [Online]. Available: https://www.researchgate.net/figure/Example-for-an-artificial-neural-network-with-two-input-neurons-two-hidden-neurons-and_fig1_289479445
[170] 'Activation Functions in Neural Networks [12 Types & Use Cases]'. Accessed: Oct. 03, 2023. [Online]. Available: https://www.v7labs.com/blog/neural-networks-activation-functions, https://www.v7labs.com/blog/neural-networks-activation-functions
[171] 'Keras Dense Layer Explained for Beginners - MLK - Machine Learning Knowledge'. Accessed: Oct. 03, 2023. [Online]. Available: https://machinelearningknowledge.ai/keras-dense-layer-explained-for-beginners/?utm_content=cmp-true
[172] 'Convolutional Layer - an overview | ScienceDirect Topics'. Accessed: Oct. 03, 2023. [Online]. Available: https://www.sciencedirect.com/topics/engineering/convolutional-layer
[173] 'Cost, Activation, Loss Function|| Neural Network|| Deep Learning. What are these? | by Mohammed Zeeshan Mulla | Medium'. Accessed: Oct. 09, 2023. [Online]. Available: https://medium.com/@zeeshanmulla/cost-activation-loss-function-neural-network-deep-learning-what-are-these-91167825a4de
[174] M. Kalirane, 'Gradient Descent vs. Backpropagation: What's the Difference?', Analytics Vidhya. Accessed: Oct. 09, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2023/01/gradient-descent-vs-backpropagation-whats-the-difference/
[175] 'How Activation Functions Work in Deep Learning', KDnuggets. Accessed: Oct. 03, 2023. [Online]. Available: https://www.kdnuggets.com/how-activation-functions-work-in-deep-learning.html
[176] N. Basta, 'The Differences between Sigmoid and Softmax Activation Function', Arteos AI. Accessed: Nov. 13, 2023. [Online]. Available: https://medium.com/arteos-ai/the-differences-between-sigmoid-and-softmax-activation-function-12adee8cf322
[177] G. Furnieles, 'Sigmoid and SoftMax Functions in 5 minutes', Medium. Accessed: Oct. 15, 2023. [Online]. Available: https://towardsdatascience.com/sigmoid-and-softmax-functions-in-5-minutes-f516c80ea1f9
[178] S. Patrikar, 'Batch, Mini Batch & Stochastic Gradient Descent', Medium. Accessed: Oct. 04, 2023. [Online]. Available: https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a
[179] 'What is Gradient Descent? | IBM'. Accessed: Oct. 04, 2023. [Online]. Available: https://www.ibm.com/topics/gradient-descent
[180] R. Python, 'Stochastic Gradient Descent Algorithm With Python and NumPy – Real Python'. Accessed: Oct. 04, 2023. [Online]. Available: https://realpython.com/gradient-descent-algorithm-python/
[181] 'How Does the Gradient Descent Algorithm Work in Machine Learning?' Accessed: Oct. 09, 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/10/how-does-the-gradient-descent-algorithm-work-in-machine-learning/
[182] 'What is Computer Vision? | IBM'. Accessed: Oct. 04, 2023. [Online]. Available: https://www.ibm.com/it-it/topics/computer-vision
[183] 'What are Convolutional Neural Networks? | IBM'. Accessed: Oct. 04, 2023. [Online]. Available: https://www.ibm.com/topics/convolutional-neural-networks
[184] 'Figure 3.2: An example of convolution operation in 2D 2 .', ResearchGate. Accessed: Oct. 04, 2023. [Online]. Available: https://www.researchgate.net/figure/An-example-of-convolution-operation-in-2D-2_fig3_324165524
[185] 'DEEPLIZARD Interactive Demo - Convolution Operation'. Accessed: Oct. 04, 2023. [Online]. Available: https://deeplizard.com/resource/pavq7noze2
[186] Dharmaraj, 'Zero-Padding in Convolutional Neural Networks', Medium. Accessed: Oct. 04, 2023. [Online]. Available: https://medium.com/@draj0718/zero-padding-in-convolutional-neural-networks-bf1410438e99
[187] M. Kumar, 'How Padding helps in CNN ?', Numpy Ninja. Accessed: Oct. 09, 2023. [Online]. Available: https://www.numpyninja.com/post/how-padding-helps-in-cnn
[188] 'Papers with Code - Max Pooling Explained'. Accessed: Oct. 04, 2023. [Online]. Available: https://paperswithcode.com/method/max-pooling
[189] 'Convolutional Neural Networks (CNN): Step 3 - Flattening - Blogs - SuperDataScience | Machine Learning | AI | Data Science Career | Analytics | Success'. Accessed: Oct. 04, 2023. [Online]. Available: https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening

[190] ‘What is Welding? - Definition, Processes and Types of Welds’. Accessed: Oct. 04, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-welding.aspx
[191] ‘Welding - What Is It? How Does It Work? 12 Types Explained’, Fractory. Accessed: Oct. 04, 2023. [Online]. Available: http://https%253A%252F%252Ffractory.com%252Ftypes-of-welding-processes%252F
[192] H. Hultman, S. Cedergren, K. Wärmefjord, and R. Söderberg, ‘Predicting Geometrical Variation in Fabricated Assemblies Using a Digital Twin Approach Including a Novel Non-Nominal Welding Simulation’, *Aerospace*, vol. 9, no. 9, Art. no. 9, Sep. 2022, Accessed: Oct. 31, 2023. [Online]. Available: https://www.mdpi.com/2226-4310/9/9/512
[193] D. Tanmoy, ‘Resistance Spot Welding: Principles and Its Applications’, in *Engineering Principles - Welding and Residual Stresses*, IntechOpen, 2022. doi: 10.5772/intechopen.103174.
[194] ‘Resistance Spot Welding (RSW) | American Welding Society Education Online’. Accessed: Oct. 04, 2023. [Online]. Available: https://awo.aws.org/glossary/resistance-spot-welding-rsw/
[195] ‘What is Resistance Welding : RWMA : American Welding Society’. Accessed: Oct. 04, 2023. [Online]. Available: https://www.aws.org/rwma/page/resistance-welding
[196] ‘What is Spot Welding? (A Complete Welding Process Guide) - TWI’. Accessed: Oct. 04, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-spot-welding
[197] ‘What is resistance seam welding?’ Accessed: Oct. 04, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/faq-what-is-resistance-seam-welding.aspx
[198] ‘Seam Welding Wheels IN-STOCK and FOR SALE | Call 844-974-9353!’ Accessed: Oct. 04, 2023. [Online]. Available: https://wsiweld.com/supplies/consumables/seam-welding-wheels/
[199] J. Grill, ‘Arc Welding Explained: What Is It & How Does It Work?’, Weld Guru. Accessed: Oct. 04, 2023. [Online]. Available: https://weldguru.com/what-is-arc-welding/
[200] ‘What is Arc Welding? - Definition and Process Types’. Accessed: Oct. 04, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/what-is-arc-welding.aspx
[201] ‘Arc Welding (AW) | American Welding Society Education Online’. Accessed: Oct. 04, 2023. [Online]. Available: https://awo.aws.org/glossary/arc-welding-aw/
[202] ‘Different Types of Arc Welding: Processes & Benefits’, Taylor Studwelding. Accessed: Oct. 04, 2023. [Online]. Available: https://www.taylor-studwelding.com/blog/types-of-arc-welding
[203] Admin, ‘What is Arc Welding? How Arc Welding Works?’, The Welding Master. Accessed: Oct. 04, 2023. [Online]. Available: https://www.theweldingmaster.com/what-is-arc-welding-how-arc-welding-works/
[204] ‘What is Friction Stir Welding (FSW)? - Process and Applications’. Accessed: Oct. 04, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/faq-what-is-friction-stir-welding.aspx
[205] D. N. Kuritsyn, V. V. Kuritsyna, and M. V. Siluyanova, ‘Digital Twins in the Design of Tools for Friction Stir Welding’, *Russ. Eng. Res.*, vol. 41, no. 4, pp. 357–359, 2021, doi: 10.3103/S1068798X21040146.
[206] RedazioneMU, ‘Friction Stir Welding: la saldatura proiettata verso il futuro’, Meccanica News. Accessed: Oct. 04, 2023. [Online]. Available: https://www.meccanicanews.com/2022/07/06/friction-stir-welding-la-saldatura-proiettata-verso-il-futuro/
[207] B. Eren, M. A. Guvenc, and S. Mistikoglu, ‘Artificial Intelligence Applications for Friction Stir Welding: A Review’, *Met. Mater. Int.*, vol. 27, no. 2, pp. 193–219, Feb. 2021, doi: 10.1007/s12540-020-00854-y.
[208] ‘Laser Welding: Definition, How it Works, Process, Types, and Advantages | Xometry’. Accessed: Oct. 16, 2023. [Online]. Available: https://www.xometry.com/resources/sheet/laser-welding/
[209] ‘What is Laser Welding and How Does it Work?’ Accessed: Oct. 16, 2023. [Online]. Available: https://www.twi-global.com/technical-knowledge/faqs/faq-how-does-laser-welding-work.aspx
[210] W. Guo, D. Crowther, J. Francis, A. Thompson, Z. Liu, and L. Li, ‘Microstructure and mechanical properties of laser welded S960 high strength steel’, *Mater. Des.*, vol. 85, pp. 534–548, Nov. 2015, doi: 10.1016/j.matdes.2015.07.037.
[211] A. Papacharalampopoulos, P. Foteinopoulos, and P. Stavropoulos, ‘Integration of Industry 5.0 requirements in digital twin-supported manufacturing process selection: a framework’, *Procedia CIRP*, vol. 119, pp. 545–551, Jan. 2023, doi: 10.1016/j.procir.2023.06.197.
[212] A. Aminzadeh, S. S. Karganroudi, M. S. Meiabadi, D. G. Mohan, and K. Ba, ‘A Survey of Process Monitoring Using Computer-Aided Inspection in Laser-Welded Blanks of Light Metals Based on the Digital Twins Concept’, *Quantum Beam Sci.*, vol. 6, no. 2, 2022, doi: 10.3390/qubs6020019.
[213] ‘Scopus - Document search’. Accessed: Oct. 05, 2023. [Online]. Available: https://www-scopus-com.ezproxy.biblio.polito.it/search/form.uri?display=basic&zone=header&origin=#basic

[214] X. Wang, Y. Hua, J. Gao, Z. Lin, and R. Yu, 'Digital Twin Implementation of Autonomous Planning Arc Welding Robot System', *Complex Syst. Model. Simul.*, vol. 3, no. 3, pp. 236–251, Sep. 2023, doi: 10.23919/CSMS.2023.0013.
[215] J. Liu *et al.*, 'Digital twin model-driven capacity evaluation and scheduling optimization for ship welding production line', *J. Intell. Manuf.*, 2023, doi: 10.1007/s10845-023-02212-2.
[216] S. Ren, Y. Ma, N. Ma, Q. Chen, and H. Wu, 'Digital Twin for the Transient Temperature Prediction During Coaxial One-Side Resistance Spot Welding of Al5052/CFRP', *J. Manuf. Sci. Eng. Trans. ASME*, vol. 144, no. 3, 2022, doi: 10.1115/1.4052130.
[217] X. Bao, L. Chen, W. Yang, and Y. Zheng, 'Integration of Digital Twin and Machine Learning for Geometric Feature Online Inspection System', presented at the IEEE International Conference on Automation Science and Engineering, 2021, pp. 746–751. doi: 10.1109/CASE49439.2021.9551440.
[218] P. Stavropoulos, A. Papacharalampopoulos, V. Siatras, and D. Mourtzis, 'An AR based Digital Twin for Laser based manufacturing process monitoring', presented at the Procedia CIRP, 2021, pp. 258–263. doi: 10.1016/j.procir.2021.09.044.
[219] H. Guo, Y. Zhu, Y. Zhang, Y. Ren, M. Chen, and R. Zhang, 'A digital twin-based layout optimization method for discrete manufacturing workshop', *Int. J. Adv. Manuf. Technol.*, 2021, doi: 10.1007/s00170-020-06568-0.
[220] H. Z. Senkara Jacek, *Resistance Welding: Fundamentals and Applications, Second Edition*, 2nd ed. Boca Raton: CRC Press, 2012. doi: 10.1201/b11752.
[221] J. Senkara, H. Zhang, and S. J. Hu, 'Expulsion prediction in resistance spot welding', *Weld. J.*, vol. 83, pp. 123-S, Apr. 2004.
[222] 'Resistance Weld Help | Problem (Expulsion / Burn Through) - Resistance Welding Supplies'. Accessed: Oct. 07, 2023. [Online]. Available: https://www.resistanceweldsupplies.com/weld-help/issues/expulsion-burn-through.html
[223] Y.-J. Xia, Z.-W. Su, Y.-B. Li, L. Zhou, and Y. Shen, 'Online quantitative evaluation of expulsion in resistance spot welding', *J. Manuf. Process.*, vol. 46, pp. 34–43, Oct. 2019, doi: 10.1016/j.jmapro.2019.08.004.
[224] 'J-Tech@PoliTO Advanced Joining Technologies', J-Tech. Accessed: Oct. 15, 2023. [Online]. Available: https://www.j-tech.polito.it
[225] 'Python os Module'. Accessed: Oct. 06, 2023. [Online]. Available: https://www.w3schools.com/python/module_os.asp
[226] J. A. Clark (Alex), 'Pillow: Python Imaging Library (Fork)'. Accessed: Oct. 06, 2023. [Online]. Available: https://python-pillow.org
[227] S. Manna, 'K-Fold Cross Validation for Deep Learning using Keras', The Owl. Accessed: Oct. 15, 2023. [Online]. Available: https://medium.com/the-owl/k-fold-cross-validation-in-keras-3ec4a3a00538
[228] 'TensorFlow'. Accessed: Oct. 15, 2023. [Online]. Available: https://www.tensorflow.org/?hl=it
[229] 'TensorFlow Datasets'. Accessed: Oct. 15, 2023. [Online]. Available: https://www.tensorflow.org/datasets?hl=it
[230] K. Team, 'Keras documentation: About Keras'. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/about/
[231] K. Team, 'Keras documentation: Keras layers API'. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/
[232] 'scikit-learn: machine learning in Python — scikit-learn 1.3.1 documentation'. Accessed: Oct. 15, 2023. [Online]. Available: https://scikit-learn.org/stable/
[233] 'NumPy'. Accessed: Oct. 15, 2023. [Online]. Available: https://numpy.org/
[234] 'Matplotlib — Visualization with Python'. Accessed: Oct. 15, 2023. [Online]. Available: https://matplotlib.org/
[235] J. Brownlee, 'How to Get Reproducible Results with Keras', MachineLearningMastery.com. Accessed: Oct. 15, 2023. [Online]. Available: https://machinelearningmastery.com/reproducible-results-neural-networks-keras/
[236] K. Team, 'Keras documentation: Python & NumPy utilities'. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/utils/python_utils/
[237] 'Stratified KFold Tutorial | AnalyseUp.com'. Accessed: Oct. 15, 2023. [Online]. Available: https://www.analyseup.com/python-machine-learning/stratified-kfold.html
[238] K. Team, 'Keras documentation: The Sequential model'. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/guides/sequential_model/
[239] K. Team, 'Keras documentation: Rescaling layer'. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/preprocessing_layers/image_preprocessing/rescaling/

[240] K. Team, ‘Keras documentation: RandomFlip layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/preprocessing_layers/image_augmentation/random_flip/
[241] K. Team, ‘Keras documentation: RandomZoom layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/preprocessing_layers/image_augmentation/random_zoom/
[242] K. Team, ‘Keras documentation: Conv2D layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/convolution_layers/convolution2d/
[243] K. Team, ‘Keras documentation: MaxPooling2D layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/pooling_layers/max_pooling2d/
[244] K. Team, ‘Keras documentation: Flatten layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/reshaping_layers/flatten/
[245] K. Team, ‘Keras documentation: Dense layer’. Accessed: Oct. 15, 2023. [Online]. Available: https://keras.io/api/layers/core_layers/dense/
[246] ‘Activation Functions for Output Layer in Neural Networks’. Accessed: Oct. 15, 2023. [Online]. Available: https://www.enjoyalgorithms.com/blog/how-to-choose-activation-function-for-output-layer
[247] ‘How does epoch affect accuracy?’, Deepchecks. Accessed: Nov. 03, 2023. [Online]. Available: https://deepchecks.com/question/how-does-epoch-affect-accuracy/
[248] ‘Why would test accuracy fall when using more epochs for training a CNN (Python, TensorFlow, Keras)?’, Quora. Accessed: Nov. 03, 2023. [Online]. Available: https://www.quora.com/Why-would-test-accuracy-fall-when-using-more-epochs-for-training-a-CNN-Python-TensorFlow-Keras
[249] ‘How does Epoch affect Accuracy in Deep Learning Model?’, GeeksforGeeks. Accessed: Nov. 03, 2023. [Online]. Available: https://www.geeksforgeeks.org/how-does-epoch-affect-accuracy-in-deep-learning-model/
[250] M. Ferdjouni, ‘Overfitting VS Data Leakage in Machine Learning’, Analytics Vidhya. Accessed: Nov. 27, 2023. [Online]. Available: https://medium.com/analytics-vidhya/overfitting-vs-data-leakage-in-machine-learning-ec59baa603e1
[251] K. Team, ‘Keras documentation: EarlyStopping’. Accessed: Nov. 06, 2023. [Online]. Available: https://keras.io/api/callbacks/early_stopping/
[252] American Welding Society (AWS): Committee on Definitions and Symbols, ‘Standard Welding Terms and Definitions Including Terms for Adhesive Bonding, Brazing, Soldering, Thermal Cutting, and Thermal Spraying’. Available: https://pubs.aws.org/Download_PDFS/1__A3_0M_A3_0_2020-PV.pdf