
SAPIENZA
UNIVERSITÀ DI ROMA

**Sapienza University of Rome**

Facoltà di Ingegneria Civile e Industriale
Dipartimento di Scienze di Base Applicate all'Ingegneria

Thesis For The Degree Of Doctor Of Philosophy

# SAPIENT: Semantic and Automatic Processing of Information about Environment

Thesis Advisor
**Prof. Domenico Vitulano**

Candidate
**Eleonora Ammaturo**

**Academic Year 2024-2025 (XXXVII cycle)**


La borsa di dottorato è stata cofinanziata con risorse del
Programma Operativo Nazionale Ricerca e Innovazione 2014-2020, risorse FSE REACT-EU
Azione IV.4 “Dottorati e contratti di ricerca su tematiche dell’innovazione”
e Azione IV.5 “Dottorati su tematiche Green”

*To Lewis Carroll,*

*for continuing to inspire us.*

## Abstract

The aim of the project, in collaboration with the company *Expert.ai*, is to implement a new system called *Sapient* which stands for *'Semantic and Automatic Processing of Information about Environment'*.

The project grew out of the need to process a multitude of complex documents, i.e. those containing information in multiple objects such as graphs, tables, etc. For this purpose, it is necessary to segment complex texts into homogeneous areas i.e. to identify the different parts that make up a document with its relative location.

This system is part of a broader one able to recognise the characters of a document and is known as Optical Character Recognition (OCR). The analysis of the structure of a document by classifying it into its components such as title, figures, tables, main text etc. is of great importance and is the main objective of the project. In the Literature this topic is known as Document Layout Analysis (DLA).

This project operates in the area of computer vision and specifically of pattern recognition in as much as documents are generally in PDF format and thus more related to an image than a text document. For the purposes of the system, the objective is not only to classify and locate the components of a text, but also to segment each component so that it can be extracted in an orderly manner. Therefore, Semantic Segmentation appears to be the best model for this purpose. In fact, it is not just an object detection problem, which is the mere identification and localisation of the document components within the same image, but also the capacity to classify the image pixel by pixel.

The classification pipeline is initially divided into two consequential steps: layout analysis and text-only analysis. For the solution of the first phase, an end-to-end Convolutional Neural Network (CNN) implementing dilated convolution is used, while for the second phase, an end-to-end multi-scale CNN is used; a heuristic within the framework of mathematical morphology is also defined for the same purpose. Finally, the segmentation of all classes simultaneously was achieved by means of another end-to-end CNN model.

The final classification allows for the segmentation of both the text and the non-text parts, thus having a final breakdown of the document into: all text parts, tables and images for non-text components and title, authors, abstract, paragraphs and its title, header, footer, notes, caption and finally lists for the segmentation of the text alone. The same classes are found in the simultaneous segmentation of text and non-text components.

The comparison with the vast Literature available, explains how this system describes an alternative overall model for DLA.



Keywords: Optical Character Recognition, Document Layout Analysis, Semantic Segmentation, Convolutional Neural Network, Mathematical Morphology.

# Contents

# List of Figures

# List of Tables

# Introduction

The last few decades have seen a surge in automated systems, commonly referred to as *Artificial Intelligence* because they perform so well that they can be compared to some human cognitive abilities. Reading is a typical ability of the human mind but in some cases it can also be performed by automatic systems.

Such systems are capable of recognising the characters of a text and extracting them from a generic document and this is known as Optical Character Recognition (OCR).

For this purpose, the extraction and organisation of information taken from written texts in structured and unstructured formats as studied in Document Layout Analysis (DLA) is a key field in the context of OCR.

The need to manipulate large volumes of documents has become increasingly relevant whatever they may be: old scanned texts as much as computer-generated files and layout analysis plays a crucial role in determining the structure and the relationship between the various visual and textual elements present in the document. DLA is part of OCR systems as the latter represent the technology for the automatic extraction of a text document into digital format. OCR systems have as their main objective the recognition of characters and their transcription into digital format. DLA deals with analysing the structure and visual organisation of the document, such as the arrangement of paragraphs, columns, images, tables and other graphic elements and is therefore a fundamental phase for OCR as it gives an order to the various parts of text.

Since text documents, whether historical or generic, often report a purely aesthetically-oriented layout, inserting graphic parts or other objects of minor importance, the analysis of the layout is necessary to be able to identify and sort them. The importance of these kinds of system is, not only because it is able to simulate human reading, but above all because text analysis indirectly provides numerous applications such as the automatic processing of large quantities of documents, the entry of data from paper documents, including handwritten ones (such as invoices, receipts, passports, business cards, mail... etc.) and the web interface to paper documents.

Generally speaking, documents contain a plethora of important information, related not only to scientific papers, but also produced by companies or institutions that require such systems to undergo an automatic analysis. It is often necessary to collect and extract this information in such large quantities as to require the process to be automated. Today, many organisations around the world, both governmental and non-governmental, still use paper documents from which it is difficult to extract the data they need. In fact, the problem of going paperless is well known

to companies: the intensive use of paper reduces efficiency in terms of cost and time, and has a significant environmental impact due to deforestation, which is a co-factor in global warming. The digitisation of paper documents is a necessity and therefore a technology that can recognise and extract data from a generic document is essential.

Therefore, although the purpose is recognition and extraction of a text, there is mainly one indirect implication which is the digitisation of the text itself. This allows the document to be archived, making it easier to find. In addition, having a system capable of quickly accessing a large amount of information allows its processing by reducing the amount of time and therefore energy required compared to the manual methods used so far.

The aim of this Research project, in collaboration with the *Expert.ai* company, is to develop a system capable of segmenting complex documents, namely those with a text layout in two columns and the inclusion of tables and graphics. More specifically, the aim of the project is not only to classify and localise regions containing text, but also to segment regions containing other components, such as graphics and tables. This stage of identifying the regions of the document that represent all its components is crucial because it allows for a complete analysis of the document.

The Thesis has the following structure:

- Chapter 2: the definition of the aims and objectives of the project and the motivation behind it.
- Chapter 3: the most significant results for the project found in the Literature on Document Layout Analysis in particular in the field of Convolutional Neural Network and Mathematical Morphology.
- Chapter 4: the explanation of the algorithm used to achieve the objectives and the mathematical approaches underlying it.
- Chapter 5: the report on the acquisition and construction of the dataset for training the model. The results obtained by the model and a comparison with the ones described in the Literature. Moreover, the extent to which the project will lead to overall improvements in terms of both energy consumption and the additional results such as digitisation and dematerialisation.

# Chapter 2

# Aims and Objectives

In order to have a broader vision of the *Sapient* project, it is necessary to explain both the objectives that the project sets itself and the motivations behind it. The project is part of a broader context that sees the reduction of impacts on climate change as one of the pillars, not only in the context of the PON Research and Innovation 2014-2020 [60], but also in various directives of European political decision-making working towards the common goal of an eco-sustainable environment.

The amount of data and information is so wide that it is necessary for these to be analysed efficiently and quickly in order to automate the process and consume the least amount of resources. Any organization, especially if governmental, often needs to produce and process a large multitude of documents, such as the Environmental Impact Assessment or reference to any European law etc. This calls for an automatic system that, once it has found this documentation, can analyse it and at the same time digitise and dematerialise it. In fact, even if the documentation is already produced by a computer, what is not obvious is that it is possible to extract the information contained in it and at the same time put a definitive end to the use of paper.

The analysis of environmental, economic, political, strategic and social information becomes a complex operation that requires the collection of a lot of data which is available in many different documents, generally in PDF format. This operation is very complex in many cases, for example, when the documentation to be analysed consists of pages in which the text is reported in two or more columns or in tables and graphs, i.e. when complex layouts are presented with more emphasis on the aesthetic features rather than formal ones. In all these cases and others reported later, it's fundamental to have a good process for the partitioning of the text with the final aim of being able to sort it in a way that reduces the margin of error in the recognition of characters and their transfer into text document format.

The need for the Research Project, as previously mentioned, arose from the idea of reducing environmental impact. *Sapient* makes it possible to process a large number of documents, through partitioning, reducing paper consumption while at the same time optimising processing times and therefore energy consumption.

The following chapter explains both the needs that led to *Sapient's* conception, as well as how it operates and finally the specific objectives that it sets out to solve.

## 2.1 Environmental Considerations

It is now clear that climate change is taking place, in fact, its effects are visible both in Europe and in the rest of the world: increase in temperatures, changes in rainfall regimes, extreme weather events such as intense rainfall, droughts and heatwaves, warming of the oceans and melting glaciers are among the most detected phenomena. Scientists argue that, with high probability, the observed changes are caused by human activities as they are linked to the increasing levels of carbon dioxide and other greenhouse gases in our atmosphere produced by humans. The report of the Intergovernmental Panel on Climate Change (IPCC) [10] already in 2014 stated: 'It is extremely likely that human influence has been the dominant cause of the observed warming since the mid-twentieth century'. It is worrying to note that these changes have a major impact on both natural and environmental resources and societies, including the territories where people live, the different crops and therefore all the economies that can thrive in certain areas.

The United Nations recognises that climate change is the greatest challenge of our time and that its negative impact compromises the ability of states to implement sustainable development, therefore, the survival of many societies and the planet's biological support systems are at risk. Understanding the increasing severity of climate change on ecosystems, biodiversity and human systems and how best to reduce the negative effects requires a framework illustrating the risk that climate change entails. In the context of climate change, the risk can arise from the dynamic interactions between climate-related hazards, exposure and vulnerability of the human and ecological systems concerned.

The vulnerability of exposed human and natural systems is a component of risk and can be extremely different within communities and between societies, regions and countries, even changing over time. Adaptation plays a key role in reducing exposure and vulnerability to climate change and, in ecological systems, includes autonomous adjustments through ecological and evolutionary processes. In the most extreme case, it can be transformative by changing the fundamental attributes of a socio-ecological system in anticipation of the impact that climate change will bring.

Resilience in ecology has several meanings and adaptation is often defined with respect to resilience as the ability to return to a previous state after a disorder. More broadly, the term describes not only the ability not to change essential structures, but also the ability to transform the system itself into a new state.

Climate change is strongly focused on the interactions between ecosystems and human society that are at the basis of the emerging risks, the degradation of ecosystems and the loss of biodiversity. Human society can adapt and mitigate, while ecosystems can do the same but within certain limits, so since human society has an impact on ecosystems, it has the task of restoring and conserving them. Meeting the goals of climate-resilient development, thus supporting both human health and that of the system and the planet, requires that society and ecosystems initiate a transition to a more resilient state. Recognition of climate risks can strengthen adaptation and mitigation actions and transitions that reduce risks. In order to mitigate these risks, global agreements have been put in place. In fact recent political decisions regarding the environment have seen significant developments across various fronts.

The first milestone on this issue was the Paris Agreement [79], adopted by 196 countries at

COP21 in December 2015, held in Paris. Its goal is to limit global warming to well below 2°C, with efforts to limit it to 1.5°C above pre-industrial levels. Countries commit to nationally determined contributions (NDCs) and to increasing their ambition over time (UNEP - UN Environment Programme).

After the Paris Agreement, several other significant international agreements and initiatives have been established to further climate action and environmental protection:

- Kigali Amendment [61] to the Montreal Protocol (2016) with the purpose to reduce the use of hydrofluorocarbons (HFCs), potent greenhouse gases used in refrigeration and air conditioning.
- the Global Pact for the Environment (Proposed 2017) an initiative to establish a comprehensive international environmental treaty that consolidates principles of environmental law. It seeks to unify various environmental agreements and provide a cohesive framework for global environmental governance. Proposed by France in 2017 and endorsed by several countries and organizations, but not yet adopted as a binding treaty (UNEP - UN Environment Programme).
- UN Climate Action Summit (2019) convened to boost ambition and accelerate actions to implement the Paris Agreement.
- European Green Deal (2019) in which the European Union's plan to make Europe the first climate-neutral continent by 2050.
- Leaders' Pledge for Nature on 2020 a commitment by world leaders to reverse biodiversity loss by 2030.
- The Glasgow Climate Pact - Cop26 on 2021 [43] in which nearly 200 nations have seen their commitments under the Paris Agreement strengthened and decided to close gaps in its implementation.
- The COP27 in Sharm El Sheikh in 2022 (United Nations-climate change) saw the establishment of a loss and damage fund with the aim to assist vulnerable countries affected by climate impacts. This was a great victory for developing countries. In addition, there was a renewed commitment for the discussions to continue on NDCs and increased funding, although progress was deemed slower than necessary to reach the 1.5°C target.
- The COP28 in Dubai in 2023 in which the EU encouraged the parties to:
  - abandon progressively the use of fossil fuels in the energy sector by 2050
  - triple the renewable energy efficiency and double the rate of improvement of energy efficiency by 2030
- The COP29 in Baku where the target of 1.5°C has been reiterated with two primary points:
  - improving ambition
  - encouraging countries to increase their contributions and trasparency of action

These agreements and initiatives represent ongoing international efforts to build on the foundation laid by the Paris Agreement, addressing various facets of climate change and environmental sustainability through collective action and cooperation. To cope with climate change, a desire on the part of the European Union is the topic of promoting a move towards a paperless organisation both within private and public sectors. In fact, the transition to paperless operations in European companies is being driven by both internal efficiency goals and external regulatory pressures from the European Union. The EU has enacted several key directives and regulations to promote digitalisation and reduce paper usage in corporate and administrative processes. The following is an overview of the most significant legislative measures:

- e-Cohesion Policy [25]: as part of the 2014–2020 programming period, the EU mandated electronic data exchange systems for managing EU funds. These systems are required to facilitate administrative checks, ensure data security, and minimise the need for paper documents, which are only requested in exceptional cases
- eIDAS Regulation [26]: this regulation standardises electronic identification and trust services across EU member states, granting qualified electronic signatures the same legal standing as handwritten ones. This facilitates the adoption of paperless processes in both public and private sectors
- Electronic Invoicing [27]: the EU is moving towards mandatory electronic invoicing (e-invoicing) for businesses, aiming to reduce administrative burdens and enhance transparency in commercial transactions. While implementation varies country to country, the trend is towards wider adoption

Italy has embarked on a comprehensive journey to digitise its public administration, aiming to enhance efficiency, reduce paper usage, and improve citizen services. This transformation is guided by several strategic initiatives and legislative frameworks:

- Code of Digital Administration (CAD) [46]: the CAD serves as the cornerstone for Italy's digital transformation. It mandates the use of digital tools in public administration, including electronic documents, digital signatures, and electronic identity systems. The CAD envisions a "teleadministration" model, where administrative procedures are conducted electronically, promoting transparency and efficiency
- Three-Year Plan for Information Technology in Public Administration [1]: published by the Agency for Digital Italy (AgID), this strategic document outlines the roadmap for digital transformation in the public sector. Key objectives include the adoption of a cloud-first approach modernising IT infrastructure in order also to enhance the interoperability between public administration systems and moreover, improving digital services for citizens and businesses.
- Public Administration Cloud (PA Cloud) [2]: AgID's cloud strategy aims to consolidate and rationalise public administration data centers by migrating services to the cloud. This initiative seeks to improve service levels and at the same time open the market to small and medium-sized enterprises (SMEs).

- Public Administration Reform under the National Recovery and Resilience Plan [24] in which Italy is undertaking a comprehensive reform of its public administration. The main aim is to simplify rules and procedures to better serve citizens and businesses.

These initiatives collectively represent Italy's commitment to a paperless and digitally empowered public administration. By leveraging technology and fostering digital skills, Italy aims to enhance service delivery and to promote transparency within the European context.

## 2.2 Optical Character Recognition system

In order to better understand the specific objectives of the Project, it is necessary to give a description of the areas in which it operates with a purely illustrative purpose without therefore entering into the merits of the techniques used. In fact, the partitioning of a text document that is the aim of the Project is part of a broader system known as Optical Character Recognition (OCR) which has the objective of recognising the characters of a text. OCR is the process of scanning handwritten or printed text to recognise the corresponding graphic characters.

Since such data is extracted and reported in a text-extension document, it is also widely used as data entry for documents of any type such as passports, invoices, bank statements, receipts and others. In fact, the indirect result is the digitisation of printed text so that it can be edited, searched, stored more compactly, viewed online and used in processes such as machine translation, text-to-speech, key data and text mining.

Literature suggests [21, 80] that the first patent for OCR was obtained by Tauscheck in 1929 in Germany. He built a machine able to read characters and numerals. The idea behind his machine was an optical and mechanical template matching and it was able to recognise characters on paper. However, it was not until 1965 that IBM brought out the first commercial OCR machine [44], 'IBM 1287', which was able to read also handwritten numbers.

From a historical point of view, these precursors were of great importance but the arrival of the computer age changed everything. In fact, it has become possible to implement methods which are more computationally complex. Today, there are many open source OCRs available to any user. Examples of these are listed below to show how they have evolved over time. An example is Tesseract [75], released in 1995, uses purely mathematical tools (such as spline functions for baseline determination or space measurement to determine lines of text) that, step by step, are able to recognise the characters in a text and extract them.

The development of deep learning in many fields, such as object detection with YOLO model [67], has led to the use of these systems, which perform better than their predecessors [82]. These systems have been extensively developed and require multiple models, even within the field of deep learning. One example is the open-source MMOCR system [50]. This describes a complete pipeline for text detection and recognition, including all the necessary intermediate steps using 14 deep learning algorithms.

As far as the Project is concerned, its aim is to break down a generic text file into its components so that it can be sorted i.e. to identify both the objects that make up the document, such as images, graphs, tables, etc. and to classify the pure text parts according to their function such as the title, abstract, main text, etc. In fact, in order to extract characters in an orderly manner, it is necessary to implement a partition of the document in question both at layout level and at text line level to distinguish all the components of the document. In this way, the document is divided into homogeneous areas that can be analysed for any type of document, including those with complex layouts i.e. containing tables, graphics and other objects. In addition, the order of the text parts to be extracted and recognised is maintained in the output. This problem is known in the Literature as Document Layout Analysis (DLA), as the State of the Art chapter clarifies.

In the following paragraph, there is a concise description of all the phases of an OCR system,

explaining the systems mentioned above, including the segmentation phase upon which the project is focused.

### Optical Character Recognition phases

For descriptive purposes only, the steps implemented by these systems are listed below. A typical OCR system consists of several components, as shown in 2.1.

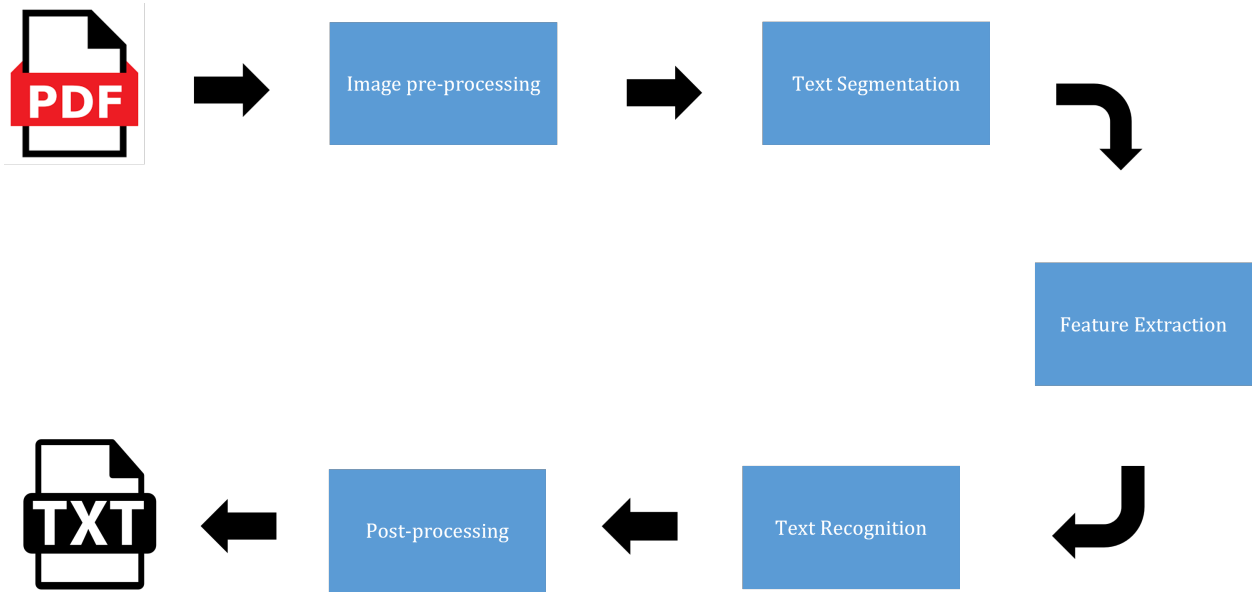


**Figure 2.1:** OCR phases

The first phase in OCR is optical scanning, this process creates an image of the paper document using a scanning device that converts light intensity into grey levels. Typically, the document is returned as a Portable Document Format (PDF) file containing all the images for each page of the document. Once the document has been captured, it is usually pre-processed: the raw data, depending on the type of capture, undergoes a series of pre-processing steps to make it usable in the subsequent stages of image analysis. In fact, the image resulting from the scanning process may contain noise, depending on the resolution of the scanner, characters may be blurred or broken and some layout components such as images etc. may not have good resolution.

Nowadays, however, software enables documents to be created and shared directly so this step can also be useful for computer-generated documents. In any case, the documents are usually in PDF format, which is processed and returned as a series of images for affinity. Errors can occur in this transformation, which is why the pre-processing phase can be necessary. These defects can cause incorrect separation of text components and errors in the subsequent character recognition stage. Therefore, they are eliminated by the pre-processing step, which, thanks to certain techniques, allows the smoothing of the digitised characters, as well as the filling, which eliminates small interruptions, gaps and holes. This pre-processing phase produces a clean character image, i.e. a normalised image with sufficient shape information, high compression and low noise. The main objectives of pre-processing can be:

- binarisation
- skew detection and correction
- noise removal
- thinning
- normalisation of the data

Then after this stage, the images of the document to be partitioned are increased in quality, and for some cases can be binary, i.e. assuming only 0 values, generally for the text parts, and 1 for the background.

The design of an OCR, as reported in [74], is de facto the definition of layout segmentation, feature extraction and text recognition steps. These processes are highly interdependent and for descriptive reasons these have been divided into stages but methods implement them organically.

The step of layout segmentation consists of the location and the identification of the various text components. It is necessary to identify the areas of the document that contain printed data and are distinct from figures, graphs or tables. This is where the Sapient project comes in. After segmentation, each symbol is extracted for regions containing text. The main objectives in segmentation are:

- identification of the graphic part or other components such as tables with the text and vice versa
- identification of paragraphs and different parts of text and text line segmentation

Since the ultimate aim of this phase is the isolation of the characters, in an orderly manner, the segmentation takes place at both layout and text level. It is in fact necessary to isolate the components relating to the written text from all parts that do not contain characters such as images, graphics etc. This leads to the partitioning of the image into areas containing text and the subsequent isolation of the individual characters. Segmentation is important because the degree of separation between the lines of the different characters directly influences the recognition performance of the individual characters. In the case of handwritten documents, segmentation serves to isolate lines and curves even in italic characters.

Therefore, as already mentioned, the ultimate purpose of such systems is to recognise the characters of a text, but since it is necessary to sort them first, the partition phase is of paramount importance. Output of this phase is then to have the image itself but partitioned, which in the specific case of text images entails a segmentation at both paragraph and text line level in order to have a sorting of characters by homogeneous blocks.

The segmentation phase is followed by the representation of each character type, that from the point of view of an OCR system represents the feature extraction and plays one of the most important roles in any recognition system. In order to minimise the complexity of the recognition step and thus to increase the accuracy of the algorithms used for character recognition, it is necessary to obtain a more compact and characteristic representation of the characters. Thus, for each class, a set of features is extracted on the basis of which it will be possible to distinguish each character from other classes, while remaining invariant to differences in features within the same class. The main

objective of the representation is to extract and select a set of features that maximises the recognition rate with the least number of elements. The aim is to obtain more representative information from the raw data, minimising the variability of patterns within the class, while improving the variability of patterns between classes. The extraction of the features has the aim of capturing the essential characteristics of symbols and is considered one of the most difficult problems in pattern recognition. The Literature is full of algorithms, also in the field of Deep Learning, that explain how this phase is carried out effectively.

The representation phase is followed by the training and recognition process that is the component of OCR system where each character is recognised thus producing a classification. This phase predominantly relies on pattern recognition, which categorizes unknown samples into predefined classes. After the training of the classification, it is then a matter of finding a match between the new character to be identified and its previous representation. For this purpose, many different techniques can be found in the Literature that identify four primary approaches to pattern recognition generally used in OCR systems:

- Template matching
- Statistical techniques
- Structural techniques
- Artificial Neural Networks (ANNs)

These approaches often overlap and techniques from one category may also be applicable in others. Template matching in OCR varies widely as it is based on the chosen feature set for representing images. Features can range from simple grayscale images to more complex graph-based character representations. Template matching can use different algorithms to classify the new character e.g. direct comparison with a set of stored prototypes and the degree of similarity is measured using metrics like Euclidean distance, decision trees, image deformation to match an unknown character with known templates by adjusting the contours of the image. The amount of deformation needed to match the two characters is used to compute a dissimilarity measure and finally, relaxation matching which involves symbolic-level image matching using feature-based descriptions. The image elements are compared to models, and the best match is found by searching in a multidimensional space. These techniques use statistical decision theory, which aims to maximize the probability of observing a given pattern based on a model of a specific class. The Literature is very extensive about these algorithms and Artificial Neural Networks (ANNs) have also been used for this purpose. This approach offers a highly parallel architecture, enabling the networks to perform computations faster than traditional methods. ANNs adapt to new data and learn the characteristics of input signals. These networks are trained with large datasets to recognise characters, and the recognition process benefits from the network's ability to learn from examples.

Finally, the last step of OCR is post-processing. Some of the most common post-processing activities are grouping and error detection and correction. Most of these are found through clustering i.e. once it has been defined which characters are assimilated, all others are eliminated. Another method of finding these errors is through the use of context i.e. defining the characters that are not relevant in that particular part of the text. This step therefore allows the OCR task to be completed to the full, eliminating everything that does not lead the system to be accurate.

After all these steps, the output document generated by OCR is a text format, so from a generic document, probably in PDF format, the result is the digitalised document.

## 2.3 Specific objectives of *Sapient*

The project *Sapient* arose from the need to implement an efficient method of partitioning any document into its components. In order to explain the objectives of the project, the description of the type of text posed for analysis is given. In fact, texts are generally divided according to their layout, and this can be subdivided, as reported in [34], into six types: rectangular, Manhattan, non-Manhattan, multi-column Manhattan, horizontal overlay and diagonal overlay.


**Figure 2.2:** Different kinds of layouts [34]

The main difference is the regularity of the layout in the position and orientation of the text.

In figure 2.2 (a) we can note a handwritten rectangular layout while in figure 2.2 (b) there is a handwritten Manhattan layout on the right and Manhattan containing multi-paragraphs on the left. Non-Manhattan layouts are those that have areas of non-rectangular shapes (see 2.2 (c)). Then, multi-column Manhattan as shown in figure 2.2 (d) commonly found in technical articles, journals, official memos etc. Finally, overlapping layouts may have one text superimposed on another and this may be due to the transparency of the initial text. These can be horizontal 2.2 (e) or diagonal overlay 2.2 (f). Non-regular layouts are characterised by texts with varying orientation and these kinds of layouts can be handwritten or typed using different styles, varied fonts and/or font sizes. These different types of layouts were identified because one branch of the analysis of document layouts concerns handwritten documents.

The project focuses on complex text documents, though not handwritten, of the multi-column Manhattan type because they contain a variety of features such as graphs, tables, images and so on. In fact, the dataset used, as explained in 5.2, contains scientific and non-scientific articles from various journals.

For the segmentation of a text document, in which the different components are distinguished, the following classes have been defined: Header and Footer, Title, Authors, Abstract, Paragraph Title, Paragraph, Lists, Notes, Graphics and Tables and their relative Captions. The figure 2.4 shows an example of a text file divided into its component parts according to the text partitioning. The image was created by overlaying a text image with its corresponding label from the dataset used to train the model and it shows the components of a generic text file and specifically: header in turquoise and footer in magenta, title in black, images in grey, authors in yellow, paragraph title in blue, abstract in red, main text in aquamarine and notes in grapecolor.


**Figure 2.3:** Example of Semantic Areas of a text image

Therefore, it is also necessary to recognise all the others classes such as Tables and Graphics etc.

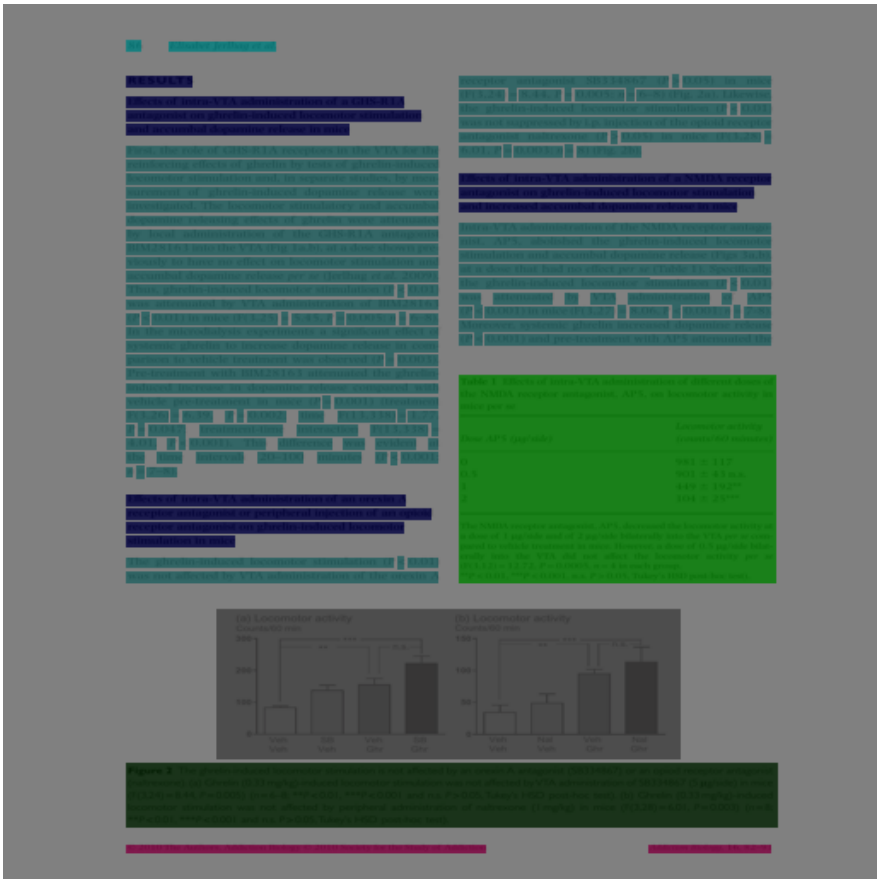

**Figure 2.4:** Example of Semantic Areas of a text image

The case shown in figure 2.5 is that of an image and its caption within a document. The purpose in this case, in addition to identifying the graph as an image, is not to consider the caption below the image as an integral part of the main text, but as something external to it, referring only to the graphic.

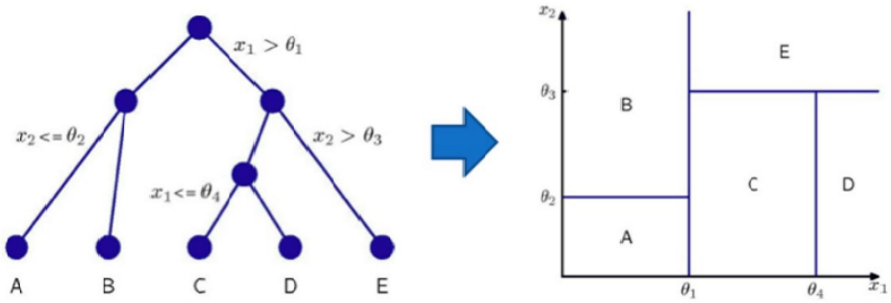


**Figure 6.** Left: generic decision-tree diagram in which two 'feature' variables $x_1, x_2$ are compared to four threshold values $\theta_i$. The result is the classification in five classes A–E. Right: the same classification represented in the corresponding Cartesian plane with the two feature variables on the axes. For a higher number of feature variables the 'tree' representation on the left is obviously easier to understand.

**Figure 2.5:** Example of a graph and its relative caption

Another aim is the recognition of tables as a separate object (an example in figure 2.6). This is in order to enable the sorting of text contained in and related to this table and not to mistakenly consider such characters as part of the preceding or following text.

**Table 1.** Number of principal components selected for the three tests by using different criteria: percentage of variance to be retained (90%, 95%, 99%), Bartlett's test with significance level $\alpha$ equal to 0.05 and 0.01, and the MDL criterion. The last column contains the expected number.

| Test | 90% | 95% | 99% | Bartlett's Test ($\alpha = 0.05$) | Bartlett's Test ($\alpha = 0.01$) | MDL | True Value |
|---|---|---|---|---|---|---|---|
| Test 1 | 1 | 1 | 1 | 30 | 30 | 5 | 5 |
| Test 2 | 37 | 52 | 75 | 90 | 90 | 90 | 3 |
| Test 2 (decimated data) | 2 | 6 | 24 | 63 | 63 | 2 | 3 |
| Test 3 | 2 | 6 | 27 | 161 | 159 | 22 | 22 |

**Figure 2.6:** Table of a scientific article

# Chapter 3

# State of the Art

The partitioning problem posed by the project goes back to a topic known as Document Layout Analysis (DLA) and focuses on the analysis of a generic text document. This problem is well documented in the Literature and has been studied extensively, thus generating a large number of results.

DLA refers to the process of analysing the structure of a document and the goal is to classify the various elements within the document such as text, images, tables, headings, paragraphs, etc. and their relationships. This analysis helps in organizing, indexing, and extracting meaningful information from documents. In fact, the purpose of Document Layout Analysis (DLA) is both to classify and locate components that belong to the physical structure of documents [34] to facilitate the subsequent recognition phases by identifying homogeneous blocks and determining the relationships between these. The ultimate aim is ordering the different components that make up the whole document. In fact, the complexity of analysing a generic document lies in the fact that these can have more aesthetically oriented layouts as well as contain many different objects such as graphs or tables.

The Literature shows that the DLA algorithms can vary depending on the document layout and the components of the final analysis. In this respect no universal DLA algorithm has yet been developed that fits all types of document layouts or satisfies all analysis components, which, as already explained, is fundamental for OCR systems. The problem has been approached in a variety of ways. However, generally it is subdivided into three main types of layout analysis strategy: bottom-up, top-down and hybrid [34]. The bottom-up strategy starts the analysis from smaller granularity data levels of an image such as pixels, characters, components, or words. It then combines homogeneous elements to form larger regions merging or grouping elements by similarity. It continues to form larger homogeneous regions until it reaches predefined stopping conditions. It estimates the parameters using statistics of pixel distributions, properties of connected components, words, text lines, or regions. The main advantage of bottom-up methods is that they are able to classify text of any shape, while the disadvantage is that they do not take into account higher-level structures in the image, such as columns, and this often leads to obtaining excessively fragmented regions. A top-down strategy, on the other hand, starts with a large region of the document and

breaks it down into smaller zones, such as columns of text, according to certain homogeneity rules. The top-down analysis stops when there are no more zones to subdivide or when certain stop conditions have been reached. Finally, the integration of both strategies (bottom-up and top-down) results in a so-called hybrid strategy.

In order to have an historical overview of DLA, it is important to refer to the International Conference on Document Analysis and Recognition (ICDAR) Challenge. The ICDAR Conference [45] is one of the leading international conferences focused on document analysis and content recognition and is a reference event for researchers and industry professionals dealing with document Image Processing, Pattern Recognition, Optical Character Recognition (OCR), Document Layout Segmentation, and other related areas. On the basis of the findings of this International Conference, the Literature presents a lot of papers. For the purposes of this project, this chapter will only report the results most relevant to the project. In fact, the following section presents the most important results found in the Literature to date. The project focuses on the classification of document components, i.e. the segmentation of the layout as well as the text itself including titles, abstracts, etc. The attention of the project, as mentioned in the objectives, has been focused on non-manuscript complex documents. There are now a large number of scientific papers in the Literature that have used Neural Networks to classify the different components of a text. In this Thesis the Literature has been divided into two main topics: Mathematical Morphology and Semantic Segmentation using convolutional Neural Network. Since there are algorithms using both techniques, the State of the Art reports also the results from these mixed methods.

This review of the State of the Art is not intended to be exhaustive, but rather to provide an overview of what has inspired the development of the project.

## 3.1 Document Layout Analysis in Convolutional Neural Networks

This section lists the Literature relating to the DLA in the field of CNN, highlighting the articles most relevant to the project and classifying them according to which components they segment.

In fact, the Literature presents numerous models based on end-to-end convolutional Neural Networks for the segmentation of the desired component.

Regarding the segmentation of the table and its contents, Table-net [63] presents an end-to-end network for both table detection and also textline segmentation therein. These are two related problems in that the latter is a subproblem of the former. For the solution, the Fully Connected Network was used, as in [57], in which the encoding branch is given by the pre-trained VGG-19 Network where the fully connected layers are replaced with two convolutional layers. Each of these convolutional layers uses the ReLU activation followed by a dropout layer. The decoding branch consists of two paths, one for the detection of the entire table and one for the segmentation of the columns within the table. So, the encoder branch is the same for both tasks i.e. the features are extracted from the whole image while the ground truth is represented by two masks, one for the whole table and the second for the internal columns. The idea is to enforce the encoding layers using as a ground truth the masks for both table and column segmentation. The paper proposes more experiments with different compositions, for the encoding path, of FCN [57] and other CNN architectures, reporting the best results for the pure Table-net model (i.e. the pre-trained VGG-19).

DeepDeSRT [72] proposes a system that divides the problem into two phases: the first detects tables within the document image, and the second recognises the structure of the tables, i.e. the identification of rows, columns and cell positions in the detected tables. Conceptually, the task of detecting tables in scanned document images is closely related to object detection in natural scene images. Based on this similarity, the paper applies domain adaptation and transfer learning techniques, leveraging deep learning-based object detection frameworks originally developed for natural scenes by selecting Faster R-CNN (FRCNN). FRCNN comprises two main components: a region proposal network (RPN), which identifies candidate regions within the input image, and a Fast R-CNN classifier, which assigns class labels to these regions. Both components share convolutional features and are trained jointly in an end-to-end manner. Because the model tended to classify all content within a table as continuous rows of pixels, a pre-processing step has been introduced in order to improve the visual separation between rows and other elements of the table. Once a table has been successfully detected and its position is known, the next challenge is to identify and localise the rows and columns that define its physical structure. To solve this task, the FCN architecture is used, which consists of applying fixed resizing factors to skip-pooled features before merging and using skip connections to merge the features of the pool2 and pool1 and pool4 and pool3 layers of the underlying VGG-16 network. To further refine the results, the system incorporates light post-processing to correct three common issues: spurious fragments, separated structures, and merged elements. The system reports F1 scores of 0.9677 for table detection and 0.9144 for table structure.

In the case of segmentation of the main text only, with the aim of excluding all other components not belonging to this, DEEP-PdF [76] uses the Unet CNN. As is well-known, especially for complex text documents such as scientific papers, a generic document contains many different objects such as images, tables, graphs and so on. The purpose of [76] is not to segment all the above-mentioned

objects from the text, but to segment the main text i.e. to take out all the text parts that don't belong to the principal paragraph, for example abstract, references etc. The network reports very good results (accuracy on the validation set of 94.32%) with a small data set, in fact it uses a dataset composed of 366 images for the training.

The paper [54] presents a model that derives from the U-net architecture [70] but it implements Trainable Multiplication Layers (TMLs) instead of the classical convolutional layers. These layers take into account the texture information of a text in order to have a more accurate segmentation of the document. Text recognition is done by considering not only the local information of each character, but also the texture of entire text regions where a high correlation is present with neighbouring pixels. Therefore, they define an end to end architecture where that of the U-Net is combined with TMLs. TML layers are placed behind the convolutional layers in order to extract co-occurring features, and by multiplying the co-occurring values at regular intervals, texture-type features can be extracted through the trained filters. The paper [54] makes a comparison between U-net and TMLs and concludes that the classification of main text and tables performs better than by just using convolutional filters.

The subsequent Literature section aims to segment multiple components of a document, often using several end-to-end CNN's in sequence, each with a limited set of objectives.

The paper [68] presents a document layout analysis segmentation of historical document images. These kinds of document present complex layouts (similar to scientific papers) that comprehend a wide variety of features that disturb layout analysis, such as multiple columns, drop capitals and illustrations, skewed or curved text lines, noise, annotations, etc. In order to segment all these different objects the system implements a pixel-wise segmentation with the U-Net architecture [70] where a ResNet-50 [38] is used for the encoder path and the decoder one follows dhsegment [4]. The model implements the CNN in several consequential stages and for some cases it is joined by heuristics. The first stage is to enhance the document itself so it implements the end to end CNN with, as a ground truth, the corresponding image with high quality. The second one is to detect the main objects and it leads to the segmentation of text regions, images, separators and background. To detect drop capitals and headings another pixel-wise segmentation has been applied. Finally, it can segment marginals with a heuristic. After the previous segmentations comes the textline detection re-implementing the end-to-end network and a heuristic to construct bounding boxes. The model can also segment vertical and mixed orientations textline (thanks to the CNN) and solve the deskewing problems by a heuristic based on the difference in pixel density due to the rotation of the text. Moreover, it can detect curve textline by identifying the outline of a text region and the corresponding text line segmentation. Finally, it sorts the reading order by top down and left to right. The comprehensive model has been compared to the State of the Art for each of the separate stages and this shows an increase in the overall weighted success rate.

The paper [88] illustrates the composition of multiple end-to-end networks for the segmentation of complex document layout and in particular segments them into: figures, tables, paragraphs, lists and paragraphs. The model proposed is a multiscale feature extraction network implementing as its main path the Resnet-101 pretrained network [38] to which it inserts two encoder-decoder branches called the MFN module. The architecture is composed of the Resnet-101 for the encoding branch and its output is sent to the MFN module i.e. the network for extracting multi-scale features. This

network, formed by two parallel branches, is applied to further process the feature maps, where one branch is represented by the Deconvolutional Pyramid Pooling Module (DPM) to enrich the multiscale information of the feature maps, while the second branch Pyramid Pooling Module (PPM) is introduced to capture the dependencies between different locations and integrate the long-range context information. At the output, the two branches are joined by the sum of the feature maps in order to improve the feature representation and thus the segmentation accuracy. This is followed by deconvolutional in order to achieve pixel-wise segmentation, the network performs upsampling to restore the feature map size. Experimental results and comparison with the Literature show that the method achieves good performance on the segmentation of figures, tables, paragraphs, lists and paragraphs. However, the complexity of the network leads to the problem of parameter compression to improve the speed of document layout segmentation.

The model [66] takes historical documents into account and aims to classify the main text, annotations, initial letter and other embellishments. It also aims to implement typographic labelling, i.e. the separation of textual content from non-textual content, and the detection of text lines and classification of highlights, main text and annotations. As is well known, training appropriate CNN's require large sets of labelled data. The article solves the problem when labelled data is limited and promotes an innovative approach that uses so-called controlled data to pre-train the networks. To this end, two strategies are proposed: the first uses artificial data, while the second uses real data to pre-train the networks. To evaluate the two strategies, several end-to-end networks are used for the detection and classification tasks, both using different variants of U-Net. The observations from the two different datasets show that, in general, the approach reduces training time while offering similar or better performance than the State of the Art.

The paper [58] presents the segmentation of generic scientific papers collecting a new dense article dataset (DAD), manually annotated, of 450 open access research articles from 14 different journals divided into three sections: front matters, body matters, and back matters, identifying 43 classes from the three sections. The dataset has been evaluated on four models that implement for the contracting path: Gated Shape CNN, FastFCN, DeepLabV3+ [11], and UNet [70], all of them utilize a ResNet50 [38] feature extractor that is pretrained on ImageNet [20]. According to the evaluation of all the models, the best network on the basis of performance takes the UNet architecture and refines its output by using it as input for a pyramid pooling module [37]. It also explains the necessary modifications to the deconvolutional branch based on the above models and uses a convolutional filter size of 5 and a dilation rate of 2. In addition, $L_2$ regularisation and dropout are used to reduce overfitting. Furthermore, in order to increase the accuracy of the segmentation results, a new boundary box regression method is also introduced. The method consists of analysing the connected components, labelling and extracting the boundary box of each object in the segmentation mask. Once the intersection over union (IoU) matrix has been calculated with the reference boxes and the predicted boxes with the greatest overlap have been selected, the gIOU loss is calculated. The cross-entropy loss of the segmentation is then treated as the classification loss, and the average of both losses is summed, with equal weight. Furthermore, to train the model on unbalanced data, loss weights are introduced for each class, which are calculated on each batch for the cross-entropy loss. Based on an evaluation of four popular segmentation models using the DAD and PubLayNet datasets, the paper demonstrates that the bounding box regression method is able to increase the $F_1$ score for DeepLabV3+ trained on DAD by +1.99 points.

In this paper [8], for historical documents, the model used is a fully convolutional network for the document layout analysis task called Doc-UFCN. The CNN relies on a U-shaped model trained from scratch for detecting objects from historical documents. The line segmentation task and more generally the layout analysis problem is achieved by the implementation of the dilated convolutional layers. The paper shows that Doc-UFCN outperforms State of the Art methods on various datasets and also demonstrates that the pre-trained parts on natural scene images are not required to reach good results, but pre-training on multiple document datasets can improve the performances. The evaluation of the models uses various metrics to reach a fair and complete comparison between the methods.

The model proposed in [83] is a multi-scale segmentation network, called MSNet, for the segmentation of four categories: text, table, figure and background. The characteristic of this CNN is its capacity to enlarge receptive field size and multi-scale feature extraction. In fact, the model implements the MS-residual module that consists in a expansion of the channel dimension In order to use separable convolutional filters in parallel with kernel size from $3 \times 3$ to $9 \times 9$. The paper shows how this model improves the evaluated performances with a fewer number of parameters.

## 3.2 Document Layout Analysis in Mathematical Morphology

In the Literature about Mathematical Morphology, it is quite common to find the pre-processing phase before the explanation of the algorithm. It is this phase which is crucial for the success of the partitioning method and depends mainly on the type of document under consideration. Therefore, the most common methods are reported. When an image pre-processing step is applied, it allows the analysis algorithms to be more robust to noise and deformation, with the purpose of correcting images or imperfections and preparing them for future high-level processing. The pre-processing phase may contain many steps such as: noise reduction, skew detection and correction, image smoothing, and skeletonisation. In addition, special attention is usually paid to the binarisation process, which aims to convert a grey-level input image into a two-level representation by dividing all the pixels of the image into two values: 0 and 1. It is the Otsu's technique that aims to convert an image from greyscale to binary i.e. composed of only two values: 0 for the background and 1 for the foreground. The method is based on the idea of finding the threshold value that minimizes the weighted variance within the class and demonstrates that this is equivalent to maximizing the variance between classes by operating directly on the histogram of grey levels. Otsu sets the threshold so as to try to minimize the overlap of the class distributions. Therefore, the method is independent of the structure of the text under analysis. A stationary statistic is assumed, although it can be modified to be locally adaptive. Thresholding to reduce a greyscale or colour image to a binary image, reduction of noise to reduce extraneous data, segmentation to separate various components in the image, and, finally, thinning or boundary detection to enable easier subsequent detection of pertinent features and objects of interest.

The Literature in the field of Mathematical Morphology for the Document Layouts Analysis is a topic that has been extensively covered and reports various algorithms.

Wong et al. [86] in their research present a method involving Run Length Smoothing Algorithm (RLSA). This method consists of two steps and aims to divide the area of a document into regions that are homogeneous with each other. The RLSA method is applied to a binary image in which white pixels are represented by a zero (0) and black pixels by a one (1). First, a segmentation procedure subdivides the area of a document into regions (blocks), each of which should contain only one type of data (text, graphic, halftone image, etc.). Next, some basic features of these blocks are calculated. This method has the effect of connecting neighbouring black areas separated by less than a constant C of pixels. The algorithm is applied row by row and column by column, resulting in two connected area maps, which are then combined with a logical AND operation. Such a block segmentation Algorithm has a limitation regarding the distance parameter chosen (C) as it is relative to the document under analysis. In fact, in different documents it can lead to connecting lines of text together that should remain separate.

Another Algorithm is the one described in [62] and presents a system that, after calculating the connected components, uses the k-nearest-neighbors (KNN) algorithm. The text document is transformed into Docstrum, a representation of the document page that describes the global structural characteristics of the page and is based on k-nearest-neighbor clustering of the connected components of the page. For each connected component, the k nearest neighbours are where the closeness is measured by the Euclidean distance between them and an angle close to zero between

them. Each pair of neighbours is then described by a tuple indicating the distance and angle between the centroids of the two components.

Chen et al. [49] propose a recursive method based on the Morphological opening operation. First the image is preprocessed to eliminate noise, then the document image is sampled at a resolution of 150 dpi to reduce the computational cost [9]. From this image an algorithm for the determination of the text line is implemented: a set of words is defined as a rectangular region containing a word. The detection of word regions is done by a pixel classifier which provides an estimate of the probability at a later date that the pixel is in a block of words or not. These probabilities are estimated from a set of training images in which the pixels have been previously labeled as belonging, or not, to a region containing words. Once the regions are identified, the system proceeds with the closing recursive algorithm where a sequence of $n$ closing transformations is performed, each characterised by a different structuring element, providing $n$ transformed images. The pixels of the training images are then used to define the vectors for the values of the corresponding positions in the transformed images, from which the posterior probabilities of these vectors being in a block of words are estimated. Therefore, a probability map is associated with the test images and a threshold value is associated to obtain the regions containing the words. This threshold value is calculated from the probability map histogram using a linear regression which gives the function between the histograms and the optimal threshold values of the images in the training set. Since the characters do not all have the same size, it is necessary that words belonging to different lines are not merged in the same block of words. In order to avoid this, a post processing based on comparison of block height and dominant block height in the document is implemented to detect these regions and divide them appropriately.

The paper [56] describes a system for segmentation and information extraction from school exercises and examination papers, addressing two main tasks: text and non-text segmentation and detection of regions of interest in examination papers. The first problem is tackled in the presence of Chinese characters, English alphabets, mathematical symbols and also complex layouts. To achieve this, a two-stage segmentation approach is used, integrating morphological transformations and Harris angles to identify small connected components (CC). The first stage allows coarse segmentation: after performing an adaptive local binarisation, due to non-uniform illumination in the scanned images, a morphological expansion operation with a $5 \times 5$ structural element is used to join connected components belonging to the same character, alleviating problems related to low image quality. The second stage allows for a refined segmentation of smaller connected components: textual lines are joined into rows, and non-textual connected components are separated from textual ones using Harris angles. A distinction is then made between text and non-text areas. In this case, the objective is to extract information from the text such as student and question information. To this end, a pre-processing phase, defined above, is performed, followed by the detection of desirable regions. To detect specific regions (such as those containing student or question information), local binarisation is used, and thin lines are removed by morphological opening. Rectangles surrounding the information are identified using polygon approximation and angle analysis to determine rectangular contours, even when these are partially incomplete due to image distortions. Once rectangles are identified, they are classified according to their geometric characteristics and relative positions on the page to extract the desired information. In summary, the proposed system integrates advanced image processing techniques to segment, identify and extract key information from both workbooks

and examination papers, addressing challenges related to image quality, layout complexity and the presence of Chinese characters.

The paper [85] presents a method for classifying headlines, text and non-text objects such as images and logos using scanned images from newspapers such as 'El Republicano' and 'El Deber'. The idea presented in this paper is to segment the document and determine the size of the main text in order to compare it with headlines and other non-text objects. The method consists of two steps: the first is a pre-processing to remove noise (paying special attention to removing the margins where there is much noise caused by the optical scanning of the document), convert the images to greyscale, improve contrast and finally make the text image binary so that the background and the text part are identified each by a single value. The second step corresponds to the layout analysis and allows the finding of structures in the document, such as paragraphs, headings and images. Once the binary image is obtained, the connected components are calculated with a connection of 8, thus determining the individual characters, each of which is delimited by a box. From these boxes, the most frequent height is calculated and the parameter $x_h$, equal to half of the most frequent height, is defined. In order to create the mask of the text to be analysed, i.e. a structure that defines the possible horizontal and vertical separators, the Morphological Opening operation is used with the square as the elementary structure. Its dimensions depend on the previously calculated parameter $x_h$. In addition, a heuristic is defined according to which a line of text is such if the ratio of the width to the height of the square is less than a threshold value. To divide paragraphs if they are arranged in several columns, the vertical outline projection is used first, then the horizontal one, otherwise the order is reversed. Therefore, if the horizontal projection is greater by a factor of 0.75 than the width of the entire document it is truncated by the same value, and similarly for the vertical projection (if this is greater by a factor of 0.75 than the total height of the document is set exactly equal to 0.75). To distinguish the main text from the headings, this is identified by bounded boxes whose height is less than those of the title or figures, and to determine the regions containing it, the dilation operation is used using an elementary square structure to connect the letters belonging to the same paragraph. Finally, the connected components are calculated and classification as main text is obtained if at least 51% of the pixels belonging to that component have an intensity below a certain threshold. To determine titles, on the other hand, after using dilation and calculating bounded boxes and related components, a proportion is used. If the ratio between the height and width of the box is greater than 0.2 and half the height of the box is greater than the parameter $x_h$, then it is considered as a title. Finally, for all elements that are not text, the following proportion is used: if the width of the bounding box is three times smaller than its height, then it is considered as non-text object.

## 3.3 Mixed Methods

*Dhsegment* [4] is a model that uses an end-to-end CNN for text extraction in ancient documents (including manuscripts) and then implements post-processing on the masks obtained through morphology operations. The architecture of the Convolutional Neural Network used is from [70] so is composed of a contracting path which in this model follows the Deep Residual Network architecture (ResNet-50 [38]) and the expanding path that is composed of five blocks plus a final convolutional layer which assigns a class to each pixel.

The post-processing phase consists of:

- thresholding with Otsu's method to obtain a binary mask (if multiclass with a multiple threshold value)
- erosion and dilatation
- calculation of connected components to filter out small ones (remove spurious pixels if there are any after the previous operations)
- vectorisation step which is required to transform the detected region into a set of co-ordinates.

In order to understand the performance of the model and to demonstrate its generality, it was applied to five different tasks relating to document analysis: the detection of an entire page, even a handwritten page, the segmentation of text lines, the detection of graphic parts, photographs and finally the analysis of document layouts. The model has the ability to both determine the desired objects and to segment the image parts of interest. In fact, if the network has the ability to classify pixel by pixel, post-processing perfects the task by constructing bounding boxes. It is the vectorisation phase that underlines the text if it is handwritten or that reframes elements that are part of the text such as graphic parts.

In [78] the model decomposes the layout analysis of a document in two subproblems: the segmentation of text and non text regions and subsequently the analysis of the only non text regions to segment tables, graphs, images etc. For the first problem the system uses two end-to-end CNN's to segment the text-line and non-text components simultaneously. For the text region segmentation the Unet architecture [70] has been used, where the encoding path implements the Resnet-34 [38]. The result of the CNN is a mask of probability values that each pixel corresponds to a class so as to remove low score pixels the Otsu filter has been applied. To get the contour of the regions, the connected components have been extracted and a rule has been defined by which a bounding box is in the set of contours if its height and weights are more than two thresholds ($T_w = 5$ and $T_h = 7$ respectively). Once all text lines have been determined, the system attempts to analyse their homogeneity in order to divide them into smaller regions. After receiving the text line components, a binary mask is generated with the foreground pixels belonging to the text line. The paragraph is defined from all closed text lines by applying consecutive morphology dilation and erosion operators on the text line mask. The elementary structure used is a vertical matrix and the height is given by the average height of the text lines. For the second phase, the segmentation of non text regions, the same CNN architecture of text segmentation has been used but here with the transfer learning techniques using pre-trained weight from ImageNet [20]. To evaluate the model a test dataset of

Vietnamese magazines has been used and the paper presents also an algorithm for textline annotations. The model was compared with the Literature on segmentation, classification and text region segmentation. The experiment results show that the method is more stable and adapts better to a new format layout.

# Chapter 4

# The Proposed Model

## 4.1 Technical Background of the proposed methodology

This chapter is dedicated to thoroughly defining the core problem addressed by this project, as well as describing the solution that has been proposed and implemented. It provides a conceptual and technical foundation upon which the subsequent work is built, aiming to guide the reader through the reasoning that underlies the development of the model. The discussion begins by introducing the concept of semantic segmentation, a widely used model for dense prediction tasks, which involves assigning a class label to each individual pixel in an image. This process enables detailed scene understanding and is essential in numerous real-world applications such as medical imaging, autonomous driving, and document analysis.

The chapter further explores how semantic segmentation is intrinsically linked to the field of deep learning, particularly through the use of Deep Neural Networks. To provide a complete context, an overview of the main principles of Deep Learning is offered, followed by a focus on convolutional Neural Networks (CNNs), which have become a standard in image processing tasks due to their ability to automatically extract hierarchical features from visual data. Special emphasis is placed on Fully Convolutional Networks (FCNs), a subclass of CNNs designed specifically for pixel-wise prediction tasks like semantic segmentation. The chapter also reviews the most influential and widely used FCN architectures that have shaped the state of the art in this domain.

Beyond neural networks, the chapter incorporates a discussion on the theoretical framework of Mathematical Morphology, a field of image processing that provides tools for analysing and processing geometrical structures within images. The fundamental morphological operations: dilation, erosion, opening, and closing are presented, along with their relevance to enhancing or refining segmentation outputs.

Finally, the complete classification pipeline developed in this project is detailed. This includes the architectural choices made for the model and the integration of morphological operations.

### 4.1.1 Semantic Segmentation with Deep Learning

Semantic Segmentation [17] is a core problem in computer vision, formulated as a dense prediction task wherein each pixel of an input image is assigned a semantic label from a predefined set of categories. As in Object Detection models, Semantic Segmentation provides both categorical and spatial information, but extends this by segmenting the parts of the image that represent the semantic of the objects requested. The output generated is a structured map that outlines objects at the pixel level. Semantic Segmentation, by definition, aims to segment semantically identified objects, thereby simultaneously solving two tasks: defining what something is and also segmenting it. Global information solves the 'what', while local information solves the 'where' and delineates it. The aim is to train a model that takes an image as input and is able to produce the same image but now partitioned, i.e. to reconstruct a sort of map defining which class each pixel belongs to. Typically the output is an image in which the classes are represented by a particular colour.


**Figure 4.1:** Semantic Segmentation: original image, original annotation (ground truth) and prediction of the model [48]

For the purposes of the project — the subdivision of a generic text document — the objective is to classify and localise each component, as well as segment them. Therefore, Semantic Segmentation seems to be the most relevant model. Literature suggests [17] this fine-grained semantic understanding is essential for numerous downstream tasks, including medical image analysis, autonomous navigation, virtual and augmented reality, video surveillance, and 3D scene reconstruction.

Early Semantic Segmentation methods primarily relied on hand-crafted features and probabilistic graphical models [31], such as thresholding techniques, k-means clustering, and conditional random fields (CRFs), which were limited in their capacity to model complex visual patterns.

The paradigm shifted with the advent of Deep Learning, particularly Convolutional Neural Networks (CNNs). The introduction of the Fully Convolutional Network (FCN) [57] marked a significant milestone, enabling end-to-end learning for dense pixel-wise prediction. Since then, CNN-based architectures have become the de facto standard, giving rise to a variety of advanced models that integrate encoder-decoder structures, multi-scale context aggregation, and attention mechanisms to further enhance segmentation performance.

### Deep Learning and Convolutional Neural Networks

The answer to the question of why Deep Neural Networks work so well has been widely discussed and one answer to this question argues that the universe is governed by a tiny subset of all possible functions [55]. In other words, when the laws of physics are written down mathematically, they can all be described by functions that have a remarkable set of simple properties. So Deep Neural Networks don't have to approximate any possible mathematical function, only a tiny subset of them. Also [81] explains that Neural Networks, in the way they learn, '*can exhibit approximate behaviour that has been described by both quantum mechanics and general relativity*'.

In this chapter we do not want to delve into these issues, however fascinating they may be, but to report Deep Learning models and especially Convolutional Neural Networks used for the purposes of the project.

As reported in [23] and in much earlier and also later Literature, it is well known that most problems cannot be modelled as linear. In fact, most data to be analysed are not linearly separable, as [71] explains: '*linear models are fundamentally limited, in the sense that by definition they cannot model non-linear relationships across features.'*

Deep Learning was born precisely to overcome this limitation by using the concept of the composition of linear problems [6].

Thus, as [71] reports: '*A powerful idea in programming is decomposition, i.e., breaking down a problem into its constituent parts recursively, until each part can be expressed in simple, manageable operations.'*

From a mathematical point of view, it is therefore a matter of defining the composition of two or more functions where each function has its own set of parameters and their composition is still a function of the same independent variable $x$ that in the field of Neural Network represents inputs data.

Since, as is well known, the composition of linear functions is still a linear function, an element of non-linearity has been introduced between the compositions. In this way, the two functions can define the sequential calculation without being merged into a single linear one. So the sequence of these functions permit the decomposition of the problem into subproblems, which can be solved by a singular function. Each of which is called a neuron and the non-linear element, an activation function.

Based on this, the definition of Multi-layer Perceptron (MLP) or Deep Feedforward Neural Network has emerged, implementing many fully connected layers. These are so-called because in each layer every neuron (unit) is connected to all those of the previous layer.

Each of these layer is defined as [71]:

$$FC(X) = \phi(XW + b)$$

where $X$ is the input, $W$ the parameters, b the bias and $\phi$ represents the non linear function.

This model is explanatory because its approximation capacity has been proven. In fact, [19, 39] demonstrate that the linear combination of compositions of a fixed function and a set of affine functions can uniformly approximate any continuous function of real variables and that they can be approximated arbitrarily well by MLPs. If the activation function is continuous, bounded and non-constant, sufficiently smooth neural networks can approximate a function and its derivatives with arbitrary precision.

The Universal Approximation of MLPs Theorem [71] is reported:

**Theorem 1** *Given a continuous function* $g : \mathbb{R}^d \to \mathbb{R}$ *exists an MLP with a single hidden layer and sigmoid activation function such that for any* $\epsilon > 0$*:* $|f(x) - g(x)| \leq \epsilon, \forall x$ *where the result holds over a compact domain.*

While the theorem makes it possible to place great importance on Neural Networks, it also takes it for granted that the depth of the network can grow unconstrained until the minimum error is obtained [7]. Moreover, the obvious practical limitation is that it is not possible to predict a priori what the best architecture is for each problem. In other words, the question of whether there is a depth that is mostly sufficient for the computation required to realise a given goal is not known a priori. Once a particular model is chosen, it is tested on different datasets on which its performance is evaluated.

As explained above, Deep Neural Networks are composed of multiple layers: input, one or more hidden layers, and output. The layers between the input and output are called 'hidden' because it is in these layers that the computational calculation takes place, i.e. the processing of the input. It is in these layers that the network's weights are calculated by composition. The great approximation capacity of Deep Learning has the downside that the processing performed in these layers is poorly accessible. Therefore, for a long time, these networks were defined as 'black boxes' and evaluated by the quality of the output obtained.

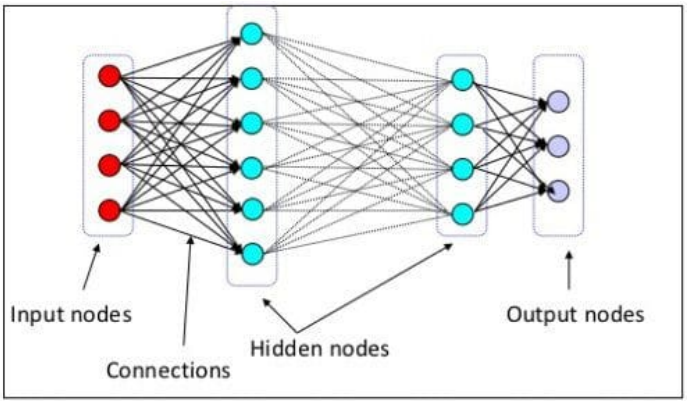


**Figure 4.2:** Deep Neural Network architecture

Such deep neural networks are called 'feed-forward' because successive layers feed forward from input to output. Therefore, the network architecture is defined once the number of layers and the number of nodes in each layer have been determined.

In the context of Supervised Learning, the Loss Function (Loss) is defined and determines the classification error and the metric used to minimise it. In fact, it is the Loss that allows the network to be trained, as in Supervised Learning the dataset is composed of input-label pairs. In this way it is possible to compare the output of the network with the label that represents the desired output according to the defined metric and the Loss that minimises the error. In the case of Deep Learning, Loss is a complicated function of composition of the weights in the previous layers. Error minimisation and therefore network weight calculation is possible thanks to the well-known *Backpropagation Algorithm* that computes the gradient of the composition function elaborated by the network. This Algorithm exploits the differential calculus chain rule, which calculates the

gradient error in terms of the summations of local gradient products on the various paths from one to the output. Although this summation has an exponential number of components (paths), it can be calculated efficiently using dynamic programming. It contains two main phases, called forward and backward phases, respectively. The advance phase is required to compute the output values and local derivatives at each node, and the products of these local values must be accumulated on all paths from the node to the output:

1. Forward phase: In this phase, the inputs for a training instance are fed into the neural network. This results in a cascade of calculations forwarded between levels, using the current set of weights. The expected final output can be compared to that of the training instance, and the derivative of the Loss Function with respect to the output is computed. The derivative of this Loss must now be calculated with respect to the weights in all layers in the backward phase.

2. Backward phase: The main goal of the backward phase is to learn the gradient of the Loss Function with respect to the different weights using the chain rule of differential calculation. These gradients are used to update the weights. Because these gradients are learned backwards, starting at the exit node, this learning process is referred to as the backward phase.

This algorithm therefore allows each epoch, i.e. an iteration of the entire dataset, to improve the calculation of error minimisation and gives the network the ability to adapt to different inputs, thus solving very different problems. As far as the use of data from images is concerned, the capabilities of Deep Learning have been integrated into Convolutional Neural Networks. These kinds of Networks, in addition to what was mentioned in this paragraph, also allow the extraction of features directly from images, which are the data input for the subsequently connected layer part.

#### Convolutional Neural Networks

The story usually recounted is that Convolutional Neural Networks evolved from studies on a cat's brain by two neuroscientists, the Nobel prize winners David Hubel and Torsten Wiesel, who investigated the functioning of visual cortex. Their experiment [42] had the aim to obtain a neural response based on what the cat's vision was subjected to. By projecting light and dark dots into the cat's eye, they were unable to get a response from the neuron. However, just by chance, they noticed that by projecting light and dark bars in different orientations, they got a response. From the experiment they deduced the existence of two types of cells in the visual cortex: 'simple' and 'complex'. The former respond best to edges or bars of a specific orientation and location and they act as feature detectors: sensitive to orientation, position, and contrast. While the latter do not respond to a particular orientation but are more spatially invariant, they respond across different positions in the receptive field and they receive input from multiple simple cells.

The Hubel and Wiesel's experiment served as the basis for the precursor of modern Convolutional Neural Networks. In order to be able to mathematically translate inferences about neurons in the cat's cerebral cortex, Convolution and Pooling operations have been taken as models. In fact, the first Convolutional Neural Network [30] implements a sequence of the simple-cells and complex-cells in a multilayer neural network architecture. The idea implemented in the model is that each convolution filter 'reads' a sub-section of the input image based on a 'preference' for a certain type

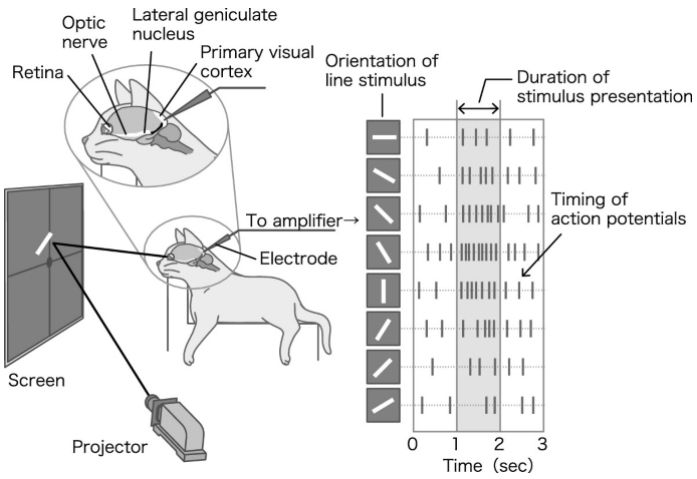


**Figure 4.3:** Hubel and Wiesel experiment [84]

of pattern, in function of the kernel used, and each of these patches is connected to another cell that fires if it gets any positive input from its corresponding patch: the pooling operation.

In order to replicate simple neurons, also called linear detectors, Convolution was used as an image filter. Convolution consists of a linear operator that is applied to each pixel of an image to change its intensity. The new intensity depends on the original intensity and the intensity of neighbouring pixels. The convolutional filter is then expressed as a matrix called a kernel.

The convolution operations and their properties are shown below. The discrete definition, as applied to images, is also provided [47]:

$$f(x,y) = (g * h)(x,y) = \sum_m \sum_n g(x-m, y-n)\, h(m,n) \tag{4.1}$$

The properties of convolution are given:

- Commutative Property: $g(x,y) * h(x,y) = h(x,y) * g(x,y)$ means that the order of convolution is doesn't influence the operation. The convolution of the input with the filter or the filter with the input arrive at the same result.
- Distributive Property: $g * (h_1 + h_2) = g * h_1 + g * h_2$ if your system is composed of multiple paths, like parallel filters, you can treat them separately and sum the results.
- Associative Property: $x(t) * \big(h(t) * g(t)\big) = \big(x(t) * h(t)\big) * g(t)$ when convolving three functions, the grouping doesn't affect the result.
- Scaling Property: $g(at) * h(at) = \frac{1}{|a|} y(at)$ compressing or stretching the input signals scales and modifies the output accordingly.

The pooling operation and its characteristics are also briefly described below. Pooling is a downsampling operation commonly used in convolutional neural networks (CNNs) to reduce the spatial dimensions of feature maps. The most common types are Max Pooling and Average Pooling.

The Max Pooling is defined as a feature map $X \in \mathbb{R}^{H \times W}$, and a pooling window of size $k \times k$, the max pooling operation with stride $s$ is defined as:

$$Y_{i,j} = \max_{(m,n) \in \mathcal{W}_{i,j}} X_{m,n}$$

where $\mathcal{W}_{i,j}$ denotes the $k \times k$ window on $X$ corresponding to position $(i, j)$ in the output $Y$.

Similarly, average pooling computes the average over the window:

$$Y_{i,j} = \frac{1}{k^2} \sum_{(m,n) \in \mathcal{W}_{i,j}} X_{m,n}$$

Global pooling reduces each feature map to a single value.

- Global Max Pooling:

$$Y = \max_{(i,j)} X_{i,j}$$

- Global Average Pooling:

$$Y = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_{i,j}$$

Pooling operations help reduce computational complexity and make the network more robust to spatial translations in the input.

The first Convolutional Neural Network as it is known today, is thanks to *LeNet-5* [53] that has well-defined characteristics [51]:

1. raw inputs
2. local receptive fields
3. shared weights
4. spatial subsampling

Convolutional networks allow images to be processed directly as input. The convolution operation operates on the image forming a sequence of convolution filters. It is these filters themselves that extract the features necessary for classification.

Convolutional filters operate on a subpart of the image in function of the kernel size, and the pixels that are involved are called receptive fields. With local receptive fields the convolution extracts elementary features which are then combined in the following layers. Distortions or shifts of the object may vary the position of the features but not the features themselves. In fact, since convolutional networks are intended to classify objects, it is essential that if the object in question is shifted or has different orientations in the image, it is in any case recognised, i.e. translation

invariance occurs. Moreover, since the convolutional kernel processes local pixels, it allows for local connectivity, so the layers are sparsely connected rather than fully connected: each node does not connect to all other nodes. Each convolutional filter processes the entire image, a process known as parameter sharing, which means that the weights to be determined are shared and therefore numerically lower than in the fully connected case.

Once the convolutional filter has processed the entire image, it is sent to a non-linear function called activation. This function introduces non-linearity into the model so that it can also learn non-linear features. As mentioned for multiple instance learning, convolutional networks introduce a composition of convolutional filters, and to ensure that this is not simply the sum of these filters, a non-linear function is inserted. These networks do not only determine a linear transformation of the input but also have the ability to extract complex features.

The subsequent pooling operation selects only certain features, such as pixels with maximum or average values. This allows the generated outputs to be summarised without regard to the location of the features, thus promoting local translation invariance. In fact, it aggregates lower-level features to produce higher-level ones, adding more invariant representation and enabling abstraction and robustness.

Since each convolutional filter is applied to the entire image, generating the so-called feature map, there is a pooling of filters for each local receptive field. The networks in fact compute many filters in parallel and thus many feature maps, but each of these is processed for the entire image and thus shared for each receptive field. Weight sharing allows for the fact that if the input of a convolutional layer is moved, also the output is moved. Furthermore, when a feature is identified, its exact position becomes less important, as long as its relative position, i.e. with respect to the other features, is preserved. Therefore, each convolutional layer is followed by a pooling layer that reduces the resolution of the feature map and the sensitivity of the output to shifts and distortions.

After the convolutional and pooling layers, the resulting features map becomes a flattened vector, which is the input for the fully connected layer or dense layers. It is in this last part that the final classification takes place based on the number of classes desired.

The *LeNet 5* network provides a good description of what has been mention so far which, in addition to implementing successive layers of convolution and pooling, uses the backpropagation algorithm to set the weights, minimising the error [53].

The document [52] laid the foundations for current convolutional networks, as it was the first network published with high performance. This network was developed by Lecun for handwritten character recognition and represents a fundamental milestone for this type of network.

The network consists of two convolutional layers, and each block consists of a convolution with a kernel size of $5 \times 5$, a sigmoid activation function, and a subsequent $2 \times 2$ average pooling operation with a stride of 2. The two layers map the spatially arranged inputs onto a series of two-dimensional feature maps, generally increasing the number of channels. The input images are $28 \times 28$ in size and the first convolutional layer has 6 output channels, while the second has 16. The output of the second convolutional block consists of 16 filters of size $5 \times 5$. To pass the output from the convolutional block to the dense block, each example in the minibatch is vectorised; this operation is called flatten vector. The idea behind the flattening layer is to convert the data into a one-dimensional array so that it can then be processed by a multilayer perceptron. The output of the last convolutional layer is then flattened into a single long feature vector, which is connected to

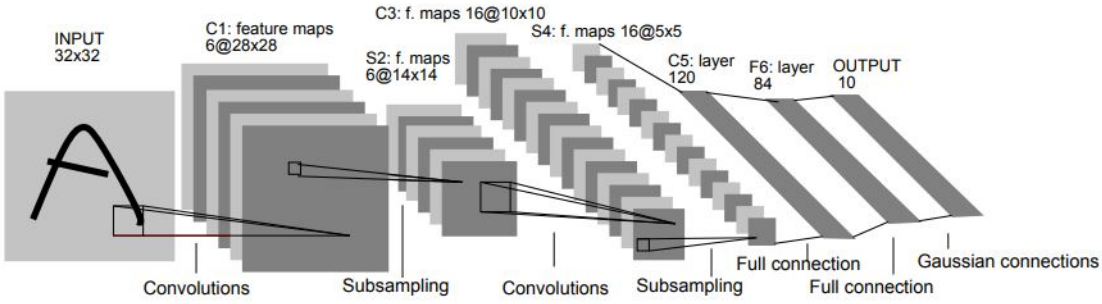


**Figure 4.4:** LeNet [53]

the final classification model, called the fully connected layer. Thus, the flattening layer converts a multidimensional array into a one-dimensional vector. Finally, each fully connected layer reduces the dimensionality, ultimately producing an output whose dimension corresponds to the number of classes.

The following is a description of the operators that make up the layers of a convolutional neural network.

**Convolutional layer:** this is the most significant component and its result is given by the convolution operation. It is thanks to the properties of convolution that translation invariance is possible.

In Convolutional Neural Networks (CNNs), 2D convolutions are essential for extracting spatial features from image data. The behavior and output dimensions of a convolutional layer are primarily controlled by three parameters: **kernel size**, **stride** and **padding** [22].

**Kernel** (or filter) is a small matrix that slides over the input feature map to compute local patterns through element-wise multiplication and summation. The *kernel size* (typically denoted as $k \times k$) determines the spatial extent of this filter, such as $3 \times 3$ or $5 \times 5$. Larger kernels capture a wider receptive field but increase computational cost.

The image below shows the 2D Convolution:

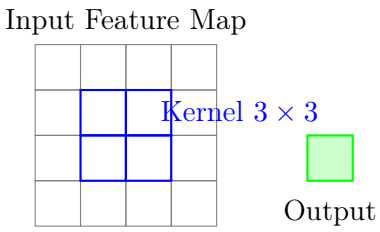


**Figure 4.5:** Visualization of a 2D convolution operation with a $3 \times 3$ kernel on a $4 \times 4$ input.

**Stride** $s$ defines the number of pixels the kernel moves after each convolution operation. A stride of 1 means the kernel moves one pixel at a time, preserving most spatial detail. Higher strides result in *downsampling*, reducing the spatial resolution of the output.

**Padding** $p$ involves adding extra pixels (usually zeros) around the input's border to control the spatial dimensions of the output. It also helps preserve edge information. Common types of

padding include:

- `'valid'`: no padding; output size shrinks.
- `'same'`: padding is added so that the output size matches the input size (for $s = 1$).

**Output Size Calculation:** given an input of height $H_{in}$ and width $W_{in}$, with a kernel size $k$, stride $s$, and padding $p$, the output height and width $(H_{out}, W_{out})$ are computed as:

$$H_{out} = \left\lfloor \frac{H_{in} + 2p - k}{s} \right\rfloor + 1$$

$$W_{out} = \left\lfloor \frac{W_{in} + 2p - k}{s} \right\rfloor + 1$$

This formulation provides a deterministic way to calculate the output dimensions of a convolutional layer, ensuring proper architectural design of CNNs.

The output of each of these kinds of layers is a collection of the input image filtered by more kernels or masks. The input image is convolved with more kernels each one generating a feature map. A kernel is de facto a matrix of weights, initially set randomly and updated during the training step. Kernels are used to find patterns in the input image and extract significant features. Kernels have three dimensions representing width, length and the number of channels (w,l,c). The function of this layer is to reduce the resolution of the image but at the same time to increase its channels.

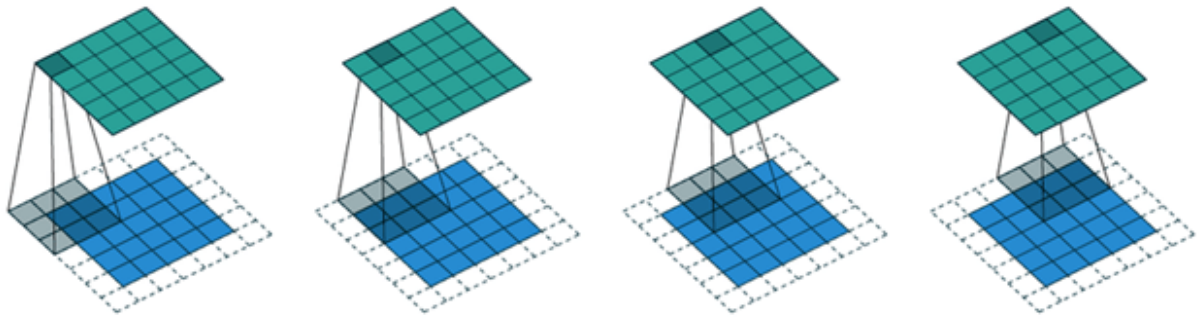

**Figure 4.6:** Convolution operation [22]

The process is an elementwise product followed by a sum. Convolution is a locally shifting invariant, which means that for many different combinations of how the nine numbers in the upper $3 \times 3$ matrix are placed, the convolution result will be the same. This invariant property plays a critical role in computer vision because for classification problems the shift or rotation of features doesn't change the object to be identified. It is worth noting that *'A horse is always a horse independent of its position in the image.'*

**Dilated convolutions:** the Convolutional layer can also introduce a spacing between kernel elements, controlled by the dilation rate $d \geq 1$. The effective kernel size becomes:

$$k_{\text{effective}} = k + (k - 1)(d - 1)$$

This allows the receptive field to grow without increasing the number of parameters, which is especially useful in semantic segmentation tasks.

The convolution operation can also be performed as a 'dilated convolution' [12, 89], these kinds of convolutions are so defined because they insert spaces between the elements of the kernel.

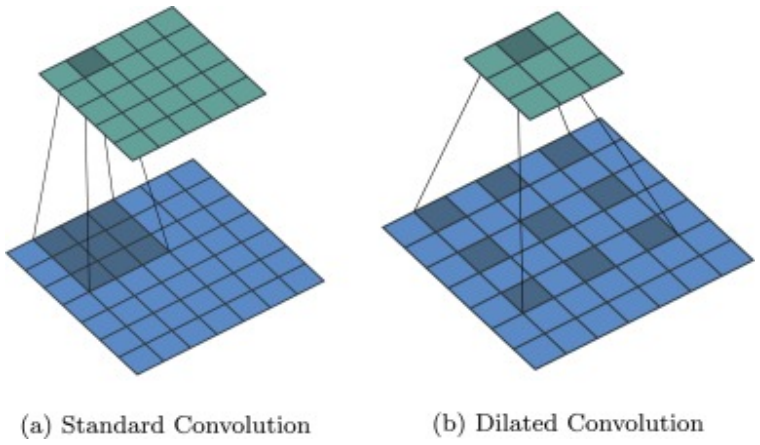


**Figure 4.7:** Standard and Dilated Convolution

The 'dilation rate' is given by an additional hyperparameter $d$ that indicates which pixels have been convolved. Implementations may vary, but usually $d - 1$ spaces are inserted between the kernel elements so that $d = 1$ corresponds to a regular convolution. Dilated convolutions are used to increase the receptive field of the output units without increasing the size of the kernel. In fact, with the same kernel size, there is an increase in the receptive field because the kernel processes pixels that are not adjacent to each other. This operation is particularly effective when several dilated convolutions are placed one after the other.

**Pooling Layer:** the main task of this layer is sub-sampling the feature maps generated by the convolutional operations [33]. This layer aims to maintain the dominant information but at the same time shrinking large-size feature maps. Several types of pooling methods are available: average pooling, minimum pooling, maximum pooling, global average pooling, global maximum pooling. In other words, the pooling layer simplifies the output by performing non-linear downsampling, thus reducing the number of parameters the network had to learn and still having an invariant property. The new methods used for pooling, include pyramid spatial pooling.

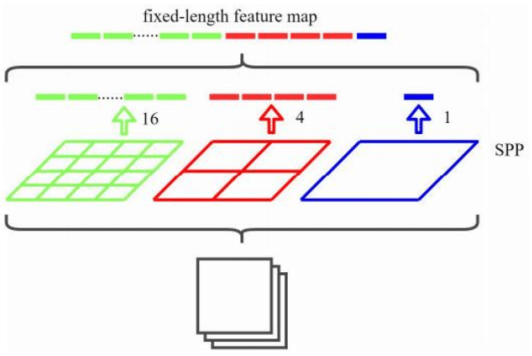


**Figure 4.8:** Spatial Pyramid Pooling [33]

This divides the image into increasingly coarse layers, aggregating the local features contained in them. This layer groups the features and generates a fixed-length output, which is then fed into the fully connected layers. In other words, it performs an aggregation of information at a deeper level of the network hierarchy between the convolutional layers and the fully connected layers.

**Fully connected (FC) Layers:** this layer is situated at the end of each CNN as it provides the actual classification. Inside this layer, each neuron is connected to all neurons of the previous layer. The input of this layer comes, as a vector, from the last pooling or convolutional layer and the output represents the final CNN output. After all the layers previously described, an activation function is employed. The operation performed by the activation function ensures that only activated features pass to the next layer. In addition, the non-linear performance of the activation layers gives to the Convolutional Neural Network (CNN) the ability to learn more complex features. The activation function must also be differentiable since it allows error back-propagation algorithm to be used to train the network.

The most known activation functions are here reported:

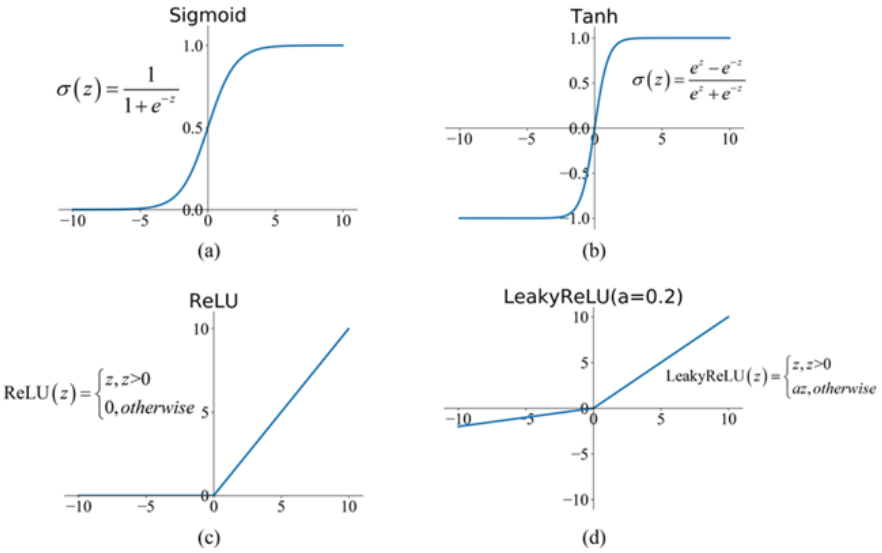


**Figure 4.9:** activation functions [29]

- Sigmoid: the input of this function is a real number, while the output is a value between

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

- Tangent Hyperbolic function is similar to the sigmoid, its input is a real number, but the output is restricted between $-1$ and 1

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

- ReLU: Rectified Linear Unit is the most used function in the CNN context. It converts the whole values of the input into positive numbers. It is less computationally expensive than tanh and sigmoid, so the computational load is the main benefit of ReLU over others (figure

2.20 (c))

$$\text{ReLU}(x) = \max(0, x)$$

- LeakyReLU: LeakyReLU activation function can be used in the convolutional layers to learn features from the input data when they assume negative values. Therefore, this activation function ensures these inputs are never ignored. Standard ReLU function would produce a gradient of 0 for these inputs and hinder the model's ability to learn (figure 2.20 d)

$$\text{LeakyReLU}(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases}$$

- Softmax: is an activation function usually used in multiclass classification problems. It is commonly used in the output layer when a probability is assigned to define the class of each input. The mathematical expression is:Softmax takes as input a vector z of K real numbers and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers. That is, before applying softmax, some vector components could be negative, or greater than one and might not sum to 1. After applying softmax, each component will be in the interval [0,1], and the sum of these probable values will be equal to 1

$$\text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

**CNNs regularisation:** regularisation is a set of strategies used to avoid overfitting, namely when a CNN learns too much from training data and fails to generalise well to new data. Due to their high complexity and a large number of parameters, CNNs can suffer from overfitting thereby reducing the accuracy and reliability of the model leading to low quality outcomes. To avoid this issue it is necessary to introduce some randomness to the learning process. There are different types of regularisation methods, but they all aim to reduce the variance of the model and increase its bias. Variance measures how sensitive the model is to small changes in the data, while bias measures how far the model is from the true relationship. A good model should have low variance and low bias, but there is usually a trade-off between them. Regularisation helps find a balance between them by shrinking or pruning the model parameters, adding noise or dropout to the layers, or augmenting the data with transformations. Some common regularisation techniques that can be used in CNNs include:

- Dropout: this technique involves randomly dropping some pixels during the training process, so they will not be a part of it. By contrast, the network is used for the prediction phase. This method forces the model to learn with different network configurations.

- Drop-weights: similar to dropout, during each training epoch, the connections between neurons (weights) are dropped.

- Data augmentation: Increasing the training dataset is the easiest way to avoid over-fitting. To generate new training samples, transformations such as rotation, scaling, and flipping are applied to the original images.

- Batch normalization: involves normalizing the inputs and outputs of each layer to reduce internal covariance shift and improve the stability and performance of the model. Subtracting the mean and dividing by the standard deviation will normalize the output at each layer reducing the "internal covariance shift". This shift becomes very high due to the continuous weight updating through training, which may occur if the samples of the training data are gathered from numerous dissimilar sources requiring extra time for convergence.

- Early stopping: implies stopping the training process when the validation loss stops improving. It works by monitoring the performance of the model on a held-out validation set for every epoch during training, and terminating if the validation performance deteriorates. This can help prevent the model from overfitting to the training data and generalizing poorly the new data.

**Optimizers:** one of the major issues included in the learning process is the learning algorithm selection, namely the optimizer chosen to minimize the training error. The most basic and most used optimization algorithm is the Gradient Descent. It's used heavily in linear regression and classification algorithms. To minimize the error, the algorithm repetitively updates the network parameters through every training epoch. Specifically, to update the parameters correctly, it needs to compute the objective function gradient (slope) by applying a first-order derivative with respect to the network parameters. Next, the parameter is updated in the reverse direction of the gradient to reduce the error. The parameter updating process is performed through network back-propagation, in which the gradient at every neuron is back-propagated to all neurons in the preceding layer. Different alternatives of the gradient-based learning algorithm are available and commonly employed. These include the following:

- Batch Gradient Descent (BGD): Batch gradient descent sums the error for each point in a training set, updating the model only after all training examples have been evaluated. This process is referred to as a training epoch. In more depth, it calculates the gradient of the whole training set and subsequently uses this gradient to update the parameters. For a small-sized dataset, the CNN model converges faster and creates an extra-stable gradient using BGD. Since the parameters are changed only once for every training epoch, it requires a substantial amount of resources, and if the dataset is too large, additional time is required for converging, and it could converge to a local optimum (for non-convex instances).

- Stochastic Gradient Descent (SGD): It's a variant of Gradient Descent. It tries to update the model's parameters more frequently. The model parameters are altered after the computation of loss on each training sample. For a large-sized training dataset, this technique is both more memory-effective and much faster than BGD. Because it is frequently updated, it takes extremely noisy steps in the direction of the answer, which in turn causes the convergence behaviour to become highly unstable. At the same time it can also be helpful in escaping the local minimum and finding the global one. To get the same convergence as gradient descent it is necessary to slowly reduce the value of the learning rate.

- Mini-batch Gradient Descent: It is an improvement on both stochastic and standard gradient descent. Training samples are partitioned into several mini-batches, in which every mini-batch can be considered an undersized collection of samples with no overlap between them. Next,

parameter updating is performed following gradient computation on every minibatch. This method requires a medium amount of memory.

- Momentum: Momentum was invented to reduce high variance in SGD. It enhances both the accuracy and the training speed by summing the computed gradient at the preceding training step, which is weighted via a factor (known as the momentum factor). However, it may simply becomes stuck in a local minimum rather than a global minimum (when there isn't a convex surface). As the value of the momentum factor becomes very low, the model loses its ability to avoid the local bare minimum. By contrast, as the momentum factor value becomes high, the model develops the ability to converge much more rapidly.

- Adam (Adaptive Moment Estimation): Adam is another optimization technique widely used for training deep neural networks. It employs the second-order derivative, and more memory efficiency and less computational power are the main advantages. The mechanism of Adam is to calculate the adaptive learning rate for each parameter in the model. The intuition behind this optimizer is to avoid jumping over the minimum but rather decrease the speed slightly for a careful search [3, 36].

### Deconvolution

In order to segment an image, the neural network must be end-to-end, i.e. it must reconstruct the input image partitioned into segmented components. It is therefore necessary, once the features have been extracted, that the network reconstructs the image in order to obtain the spatial variable of these features. As explained, convolutional networks perform a downsampling of the image to obtain the features, thus significantly reducing the resolution. In order to perform the reverse process, i.e. image reconstruction, it is necessary to define an operation that goes back from the last feature map to the convolutional steps performed up to that point. This operation is defined as deconvolution, the 'inverse' of convolution. That is to say, the deconvolution is defined as the transformation that goes in the opposite direction to a convolution, so it starts from the output of a certain feature map in order to reconstruct the input image. It arises from the need to obtain this transformation as the decoding of a convolutional network or to project feature maps into a high-dimensional space.

Various operations have been applied for deconvolution such as:

- Transpose Convolution
- Upsample
- Unpooling

**Transposed Convolution:** is one of the more common learnable techniques that allows the expansion of the spatial dimensions of feature maps [22]. In contrast to standard convolution, it distributes each input value over a larger region of the output. A learnable convolutional filter is applied to the reduced size feature maps in order to turn around the effects of the convolution operation permitting the forward and backward passes of a convolution. It is a learnable method that is optimized during training, its weights being calculated during the training phase as for convolution filters and is effective for generating high-quality images of larger size. The operation of convolution uses a kernel to filter the input or the feature map, producing a set of output features maps smaller than the input, due to the kernel size and stride. In transposed convolution instead of applying a kernel to a local region of the input, a kernel is applied to each location of the output feature map, with a stride that determines the distance between successive kernel positions. Each kernel output is then added to the corresponding location in the output feature map.

From an algebraic point of view, it is defined as the transposed convolution whose forward and backward passages are calculated by multiplying by a matrix. If the convolution operation is described as the product of the input by a matrix that determines the output: $input * W = output$, the convolution matrix could be represented as a sparse matrix W in which the non-zero elements are the elements $w_{i,j}$ of the kernel with i and j indicating the row and column of the kernel, respectively. This linear operation takes the flattened input matrix as an n-dimensional vector and produces an m-dimensional vector. Using this representation, the backward pass is easily obtained by transposing W. Thus, $output = W^T * input$. This operation takes an m-dimensional vector as input and produces an n-dimensional vector as output, via a matrix that is the transpose of the convolution matrix. This results in an output with restored dimensions and a filter whose weights can be calculated during training.

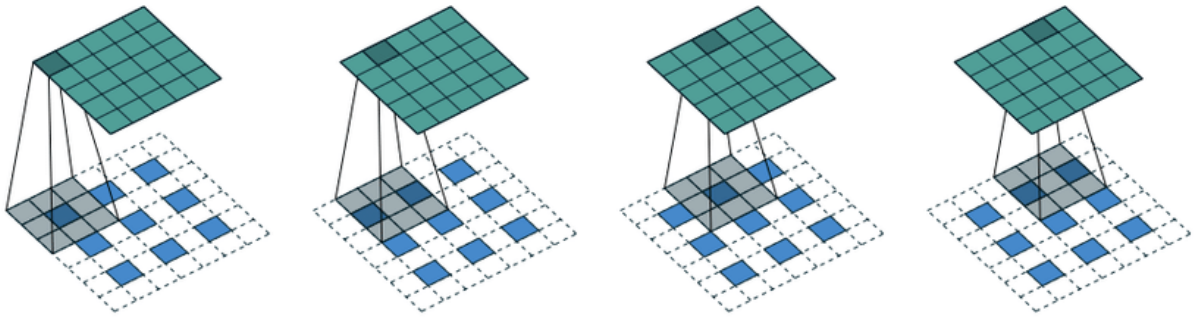

**Figure 4.10:** Transposed Convolution [22]

The image 4.10 represents a clarification of the transposed convolution operation.

**Upsampling operation:** allows images with the desired density to be reconstructed from the raw outputs, gradually restoring the spatial dimensions of the feature maps while reducing the number of channels or features.

This approach can be implemented with many different interpolation methods that learn how to refine the output:

- Nearest Neighbor: This method involves duplicating the existing pixels or samples to create new ones. It is the simplest and fastest method but the result is a block of identical pixels and therefore highly correlated.

- Bilinear Interpolation: it reconstructs a 'continuous' image from the discrete input image by performing linear interpolation in two dimensions. For a point $(x, y)$ in the upsampled grid, the corresponding coordinates $(x', y')$ in the original lower-resolution grid are found. Since these coordinates usually fall between discrete input pixels, they are interpolated between the four nearest neighbors. So, to compute the value of new pixels, the weighted average of the nearest four pixels is calculated. This results in a smoother image than nearest neighbor interpolation but may introduce artifacts such as blurring and aliasing.

- Bicubic Interpolation: This method uses a more complex interpolation algorithm that takes into account 16 neighboring pixels to compute the value of new pixels. It results in a higher-quality image than bilinear interpolation but requires more computational resources.

- Lanczos Interpolation: This method uses a weighted average of a larger number of neighboring pixels to compute the value of new pixels. It produces sharp and detailed images but requires a larger kernel size and more computational resources.

**Unpooling operation:** is defined as the inverse operation of pooling and uses the indices saved during the pooling phase to correctly reposition the values in the upscaled feature map. This method preserves the original spatial information, but requires explicit index management. As is well known, the convolutional and pooling layers of a CNN progressively reduce the spatial dimensions (width and height) of the input and intermediate feature maps, a process known as encoding or downsampling.

This produces a compressed representation of the input image that can be efficiently processed, allowing the network to extract key structural features and relationships within the input. The pooling process, as explained in 4.1.1, reduces the image resolution (generally using a pooling factor of 2, thus halving it). To reverse the process at each pooling iteration, its indices are stored, i.e. the position where the pixel value is located. When the image is doubled, the pixels are returned to the same indices for which they were reduced. The image clarifies this operation.

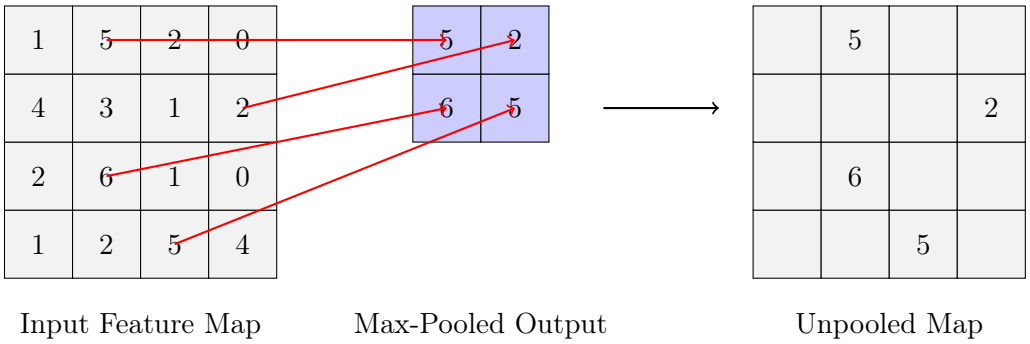


**Figure 4.11:** Visualization of Unpooling Operation

Whichever inverse convolution operation is chosen, it must be refined during the training phase. In fact, the parameters initially generated in the manner described above do not determine the desired output image, but only its resolution.

### Fully Convolutional Neural Networks

The evolution of convolutional networks as classifiers and localisers came from Fully Convolutional Networks [57]. This marks a milestone in the development of convolutional neural networks, as it modifies their architecture. As reported in 4.1.1, the architecture of a Convolutional Network is composed by a sequence of convolutional layers followed by fully connected layers. The modification is to replace the fully connected layers with convolutional layers. This is why it is known as a 'fully convolutional network'. The idea behind this being to cut out the fully connected layer for the classification task and to substitute it with convolutional layers that could classify and reconstruct the image. In this way, the spatial dimension is not lost and it is therefore possible not only to classify but also to localise objects by taking images of any size as input, unlike networks that have fixed dimensions for the input because of the fully connected layer. In fact, for CNNs designed solely for classification, the input image size must remain fixed once the network architecture has been defined. A deep network computes a general nonlinear function, a network with only convolutional layers, i.e. a fully convolutional network, computes a nonlinear filter.

Therefore, the network presents layers composed by a sequence of convolution - max pooling - activation function and these layers follow one another to define the composition of functions as for any deep learning network.

To better understand this modification, [57] explains that fully connected layers can be modified into convolutional layers with kernels of size equal to the input size of the fully connected layer.

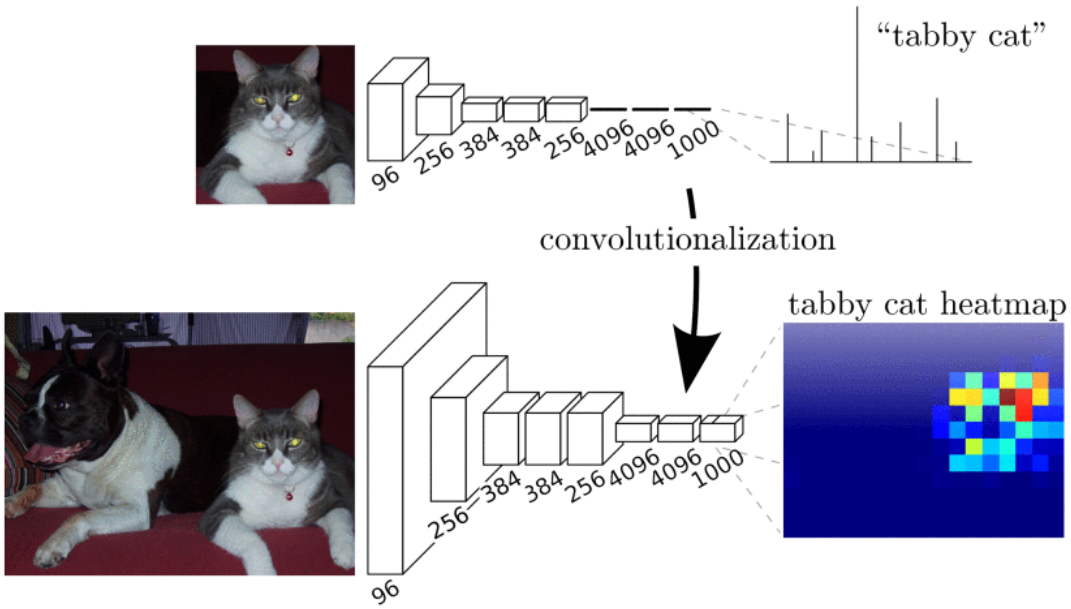


**Figure 4.12:** From Convolutional Network to Fully Convolutional [57]

For the construction of the network, in the encoding path, already known CNN were taken such as: AlexNet, VGG nets and GoogLeNet which, as [57] explains, *'We decapitate each net by discarding the final classifier layer, and convert all fully connected layers to convolutions.'* Furthermore, the major difference between fully connected and fully convolutional networks is the number of parameters. In fact, in the convolutional case, parameter sharing allows for a reduction in these.

The 'convolutionalization' is followed by the decoding path, which is composed of a deconvolution

layer, as explained in 4.1.1, that in [57] is implemented by the bilinear interpolation that can upsample the coarse outputs to pixel-dense outputs. The final prediction that occurs in the last convolutional layer usually has a very low resolution (in this network 32 pixels) thus also limiting the resolution in the upsampled output. To solve this problem the network adds links that combine the final prediction layer with lower layers with finer feature map. In addition, also in this task there is a reduction in computational cost since the resulting maps are equivalent to the evaluation of the original network on particular input patches (receptive fields), i.e. those that are 'activated' during the training process, and the calculation is significantly decreased thanks to the overlapping regions of these patches. In this way, the network produces as many maps as there are desired classes, each representing the segmentation features for that class. This transformation of the CNN into Fully Convolutional Networks permits inputs of any size and allows as output the same image after classification and localisation of the desired objects, thus composing an end-to-end network.

### Further Architectures

**The U-net** architecture derives from [57] with the modification of the deconvolution path. In fact, the main idea of this architecture [70] is to substitute pooling operators with upsampling, followed by a $2 \times 2$ convolution in order to increase the resolution of the output. If [57] modified the usual architecture of a CNN to a fully convolutional one, the U-net [70] is an evolution of this because it provides an increase of the deconvolution path creating a sequence of Upsampling convolutional layers (as Upsampling operation 4.1.1 explains). In fact, the decoding path has a large number of channels for features, enabling the propagation of the context information and ensuring higher resolution levels. Consequently, this path is symmetrical to the contraction path and so the architecture has a U-shaped form. As already mentioned, the network has no fully connected layers and uses only the valid part of each convolution, i.e. the segmentation map contains only those pixels for which the full context is available in the input image. To predict the pixels in the boundary region of the image, the missing context is extrapolated by mirroring the input image. To localise high-resolution features, the contraction path is combined with upsampling features thanks to skip connections.

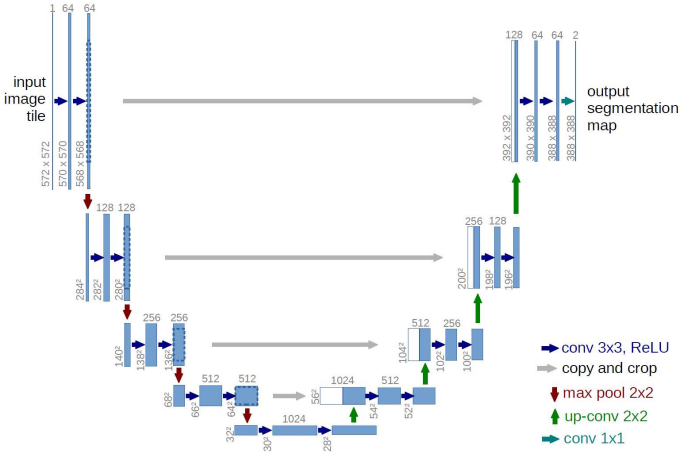


**Figure 4.13:** Unet Architecture [70]

As in the architecture of Fully Convolutional Network 4.1.1, this network is composed by a contraction and an expansion phase. The former follows the typical architecture of a convolutional network and consists of repeating $3 \times 3$ unpadded convolutions, each followed by a ReLu activation function and a $2 \times 2$ max pooling operation with stride 2. So each of these steps doubles the number of feature channels. Instead, each step in the expansion phase consists of upsampling the feature map followed by a $2 \times 2$ convolution that halves the number of feature channels. The network copies and crops each feature map of the contraction phase to the corresponding expansion one. This is necessary due to the loss of pixels at the edge in each convolution. Each convolution implemented has a $3 \times 3$ kernel and is followed by a ReLu activation function. The last layer consists of a $1 \times 1$ convolution which serves to map each 64th component of the feature vector to the desired number of classes.

**The Adaptive U-net** [59] is a variant of the original U-Net and was designed for detecting lines of text in images of historical documents [66]. The modification to the original U-Net model lies in the number of filters for each layer. In fact, the original U-Net uses 64, 128, 256, 512 and 1024 filters in each block of the contraction path, while the Adaptive U-Net model proposes the use of 32, 64, 128, 256 and 512 filters. This architecture also uses skip connections so that the features extracted from the contraction path are forwarded to the expansion path through horizontal connections. Each convolution block in the contraction path is followed by max pooling. The tensor that is passed to the expansion path is the bottleneck. The expansion path consists of four convolution blocks, each represented by two simple convolutions followed by a transposed convolution (as explained in 4.1.1).

**The Doc UFCN** [8] defines a convolutional network similar to U-net where dilated convolution is used instead of convolution. The contraction path consists of 4 dilated blocks consisting of 5 consecutive dilated convolutions. This allows the receptive field to be increased and provides the network with more contextual information. Each block is followed by a max pooling layer, except for the last one. The expansion path aims to reconstruct the input image with pixel-by-pixel labelling at the original resolution of the input image. In this network, deconvolution has been implemented with transposed convolutions in order to maintain the same resolution at both the input and output. Therefore, the decoding path consists of 3 convolutional blocks, each consisting of a standard convolution followed by a transposed convolution. In addition, the features calculated during the encoding phase are concatenated with those of the decoding phase. The last convolutional layer produces full-resolution feature maps. It returns $n$ feature maps with the same dimensions as the input image, where $n$ is the number of classes involved in the experiment. A softmax layer is then applied to transform these feature maps into probability maps.

**The DeepLab Architectures**

The first introduction of the DeepLab architecture is presented in [13]. This network proposes an innovative approach to semantic image segmentation by combining two established techniques: Deep Convolutional Neural Networks (DCNNs) and fully connected Conditional Random Fields (CRFs).

The initial part of the network follows a typical CNN structure, consisting of a sequence of convolutional layers and max-pooling operations for progressively extracting hierarchical features. However, DeepLab introduces a key innovation: *atrous* (or *dilated*) convolutions.

As explained in 4.7, dilated convolutions allow the expansion of the receptive field of convolutional filters through a dilation rate, without increasing the number of parameters or the computational cost. This is achieved by "skipping" certain pixels in the input, effectively inserting gaps—called *trous*, from the French word for "holes"— between kernel values. In this way, the context analysed by the filter is enlarged while maintaining the same output resolution.

The use of DCNNs with dilated convolutions addresses the spatial resolution loss typical of deeper CNN layers, while eliminating the need for deconvolutional stages. The output produced is a preliminary segmentation map, which, however, may lack accuracy around object boundaries. To refine this, DeepLab incorporates a fully connected CRF module capable of modeling relationships among all image pixels. In this setting, each pixel is treated as a node that receives a label based on similarity with its neighbors. CRFs are particularly effective for denoising and enhancing edge precision, iteratively refining the segmentation result from the DCNN.

The second version of the architecture, **DeepLabv2** [14], was published in 2017 by the same authors. While retaining the core structure (DCNN + atrous convolutions + CRF), it introduces a significant extension: *Atrous Spatial Pyramid Pooling* (ASPP).

This technique is designed to handle the scale variability of objects in images. ASPP applies atrous convolutions with different dilation rates over the same input region, allowing the network to gather contextual information at multiple scales. The aggregation of the resulting features improves the network's ability to distinguish objects of different sizes, thereby enhancing segmentation performance. However, like its predecessor, DeepLabv2 relies on CRFs which prevents end-to-end training of the entire model.

In December 2017, the authors introduced **DeepLabv3** [15], aimed at overcoming the limitations of previous versions. The core architecture (DCNN + ASPP) is retained, but the CRF refinement stage is removed, thus enabling fully end-to-end training. Despite this removal, the network achieves improved segmentation accuracy compared to DeepLabv2.

In addition to the parallel atrous convolutions in ASPP, DeepLabv3 also introduces *cascaded atrous convolution modules*, applied in sequence. This cascading structure enables the use of multiple convolutional layers without reducing the output resolution, thereby avoiding the loss of spatial information that occurs in traditional CNNs. By combining convolutions with varying dilation rates, the network can progressively capture broader contextual information, improving segmentation accuracy.

Whereas in cascaded structures the atrous convolutions are applied sequentially, in ASPP they are executed in parallel and their outputs are fused. Moreover, the extracted features are enriched by fusing them with other feature maps, further improving the network's ability to capture both structural and semantic details. The final output is then upsampled to reach the desired resolution.

The joint use of ASPP and cascaded atrous convolutions enables the model to preserve spatial precision while maintaining a strong grasp of contextual information.

The most recent development in the architecture, **DeepLabv3+** [16], was proposed in 2018. Compared to DeepLabv3, it introduces a decoder module aimed at further improving the resolution of segmented object boundaries. The encoder portion remains largely similar to that of DeepLabv3, but a more efficient variant of convolutions is adopted: the *Separable Atrous Convolution*.

These operations are divided into two steps: *depthwise convolution*, which applies separate filters to each input channel, and *pointwise convolution* (1×1 convolution), which combines information across channels. This separation results in a significant reduction in computational complexity without compromising segmentation quality.

The decoder has a simple structure based on upsampling operations and is responsible for recovering fine details at object boundaries, often lost during the encoding phase. The integration of the decoder allows DeepLabv3+ to achieve even more accurate results, setting new benchmarks for semantic segmentation architectures.

**The Multiscale Segmentation Network (MSNet)** [83] model for document layout analysis employs an enhanced residual block, termed *MS Residual Module*, as the fundamental building unit of the encoder backbone. The backbone is organized into four stages, with each stage performing a downsampling operation by a factor of 2. Each stage comprises two MS residual modules, resulting in a final feature map with spatial dimensions reduced to 1/32 of the original input image.

To maintain computational efficiency and limit the number of parameters, a non-local attention module after the final residual block in the backbone has been integrated. This module captures long-range spatial contextual information, which is essential for accurate layout parsing.

In the decoding phase, a U-shaped skip connection structure inspired by U-Net has been adopted, which facilitates the recovery of fine-grained details by fusing high-level semantic features with lower-level spatial information. The upsampling and classification module restores the feature map to the original input resolution, generating the final segmentation output.

The MS residual module is designed in two variants, depending on whether downsampling is required, as shown in 4.14.

Unlike traditional residual networks [38], which reduce the channel dimension before expansion, the approach first increases the number of channels using a $1 \times 1$ convolution (doubling the channel size), then partitions the feature map into four groups along the channel axis.

Each group is processed using a depthwise separable convolution [40] with different kernel sizes: $3 \times 3$, $5 \times 5$, $7 \times 7$, and $9 \times 9$, enabling multiscale feature extraction. The outputs are concatenated along the channel dimension to form a unified feature representation.

Next, a grouped $3 \times 3$ convolution [87] is applied to fuse the multiscale features. A final $1 \times 1$ convolution reduces the channel dimension back to the original size. Additionally, a Squeeze-and-Excitation (SE) channel attention module [41] is inserted to enhance informative features adaptively. The input is then added to the output via a residual connection.

During downsampling, a max-pooling operation and a $1 \times 1$ convolution are applied to the input feature map before the residual connection to ensure dimensional consistency. Batch normalization is added after each convolutional layer, and ReLU activation is used throughout the network to enhance non-linearity.

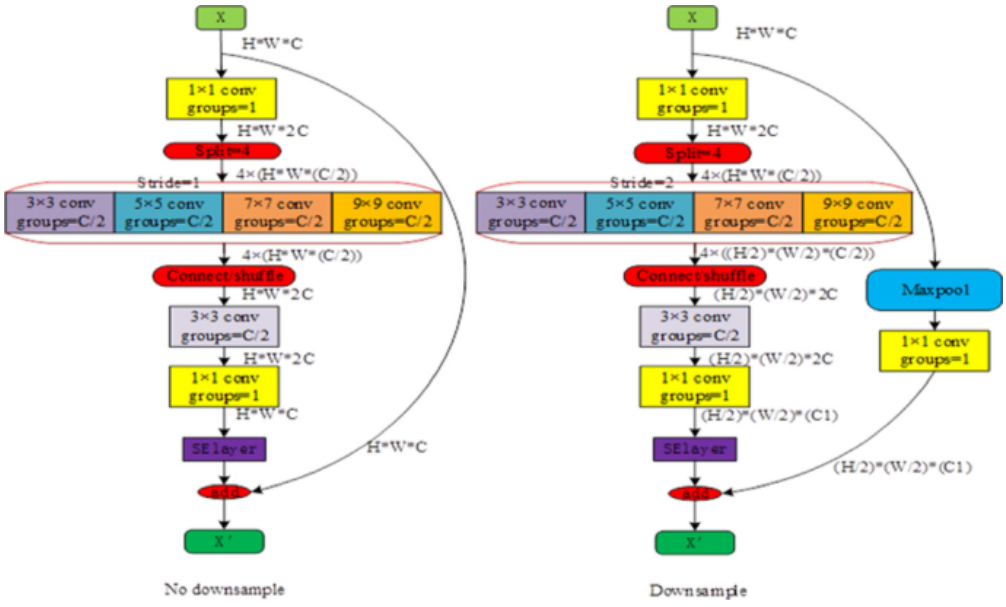


**Figure 4.14:** MsNet module [83]

**The Segnet** [5] end-to-end convolutional Neural Network architecture has been designed for road scene segmentation. It can model appearance and shape and understand the context between different classes. Consequently, it can also preserve contour information in the extracted image representation. The encoder network in SegNet is topologically identical to the convolutional layers in VGG-16 [73]. The decoder network is the key component of SegNet and consists of a hierarchy of decoders, with one corresponding to each encoder. The appropriate decoders use the max-pooling indices (explained in 4.1.1) received from the corresponding encoder to perform nonlinear upsampling of their input feature maps. Reusing max-pooling indices in the decoding process

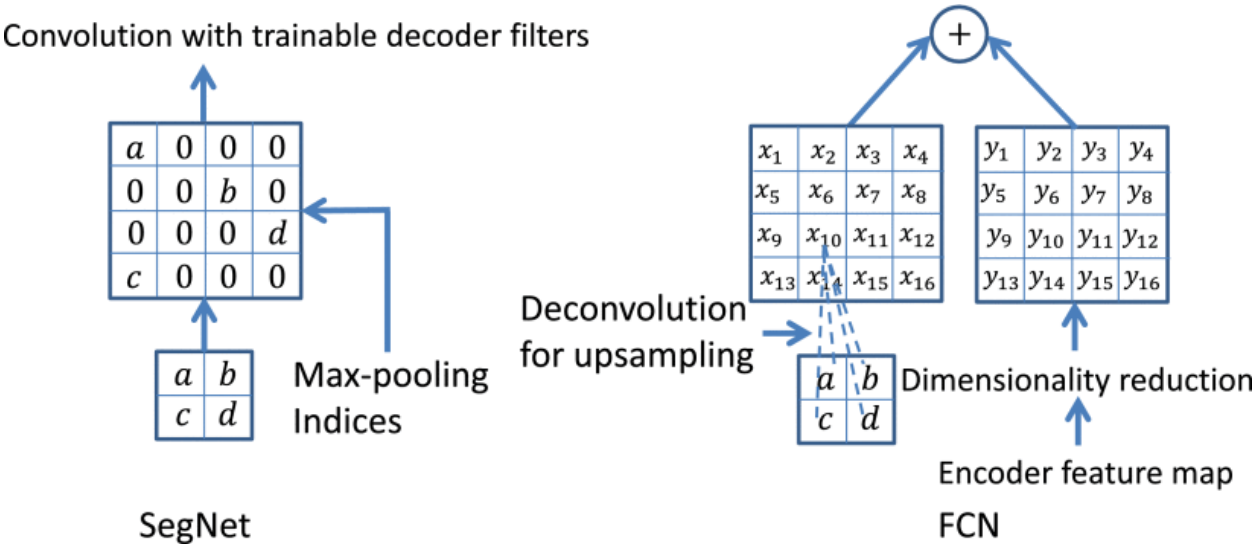


**Figure 4.15:** Segnet Upsampling [5]

offers several practical advantages: improved boundary delineation, reduced parameters, enablement of end-to-end training and the ability to incorporate this form of upsampling into any encoder-

decoder architecture. In fact, the network reduces significantly the number of parameters for the encoder path, from 134 to 14.7 million. The encoder network comprises 13 convolutional layers, which correspond to the initial 13 layers of the VGG-16 network, which was designed for object classification. So it is possibile to implement transfer learning. For each encoding layer corresponds a decoding layer, therefore, also the decoding network has 13 layers. The final output of the decoder is sent to a multi-class softmax classifier that produces class probabilities for each pixel independently.

**The PSPNet** [90], for semantic segmentation for scene parsing, was created with the aim of avoiding errors that are partially or completely related to the contextual relationship and global information for different receptive fields. Therefore, it develops a deep network with an appropriate level of global scene, i.e. to take into account three aspects:

- Mismatched Relationship: it is necessary to consider the context in which the segmenting objects occur. Indeed, the inability to collect contextual information increases the possibility of classification errors and confusion between categories. This problem can be solved by using the relationship between categories.

- Confusion categories: for the purposes of the network, many class labels can cause confusion in classification. In fact, some classes differ little from each other as they have a similar appearance (the text in a table can be similar to the one in a paragraph). Furthermore, it is plausible that even the best datasets may contain errors or biases (there is a known error of 17.60% in terms of pixels, as described in [43]). This problem can be solved by using the relationship between categories.

- Inconspicuous Classes: the unremarkable class scene contains objects of arbitrary size. This is very common in DLA, where text can have different fonts and images or tables can have different formats. To improve the performance of significantly small or large objects, special attention must be paid to the different subregions that contain unremarkable category elements.

The network is innovative as it implements the pyramid pooling module, which empirically proves to be effective in analysing the context. The aim is therefore to expand the receptive field by incorporating different levels of pooling by merging information from different sub-regions with these receptive fields. The pyramid pooling module merges features into four different pyramid scales.

The coarsest layer highlighted in red is global pooling to generate a single container output. The following pyramid level separates the feature map into different subregions and forms an aggregate representation for different locations. The output of the different levels in the pyramid pooling module contains the feature map with various dimensions. To maintain the weight of the global feature, a $1 \times 1$ convolution layer is used after each pyramid level to reduce the size of the context representation to $1/N$ of the original one if the pyramid layer size is $N$. The low-dimensional feature maps are then directly upsampled to reach the same size as the original feature map via bilinear interpolation. Finally, different levels of functionality are chained together as the final pyramid pooling global functionality. Note that the number of pyramid levels and the size of each layer can be changed. They are related to the size of the feature map that is placed in the pyramid pooling

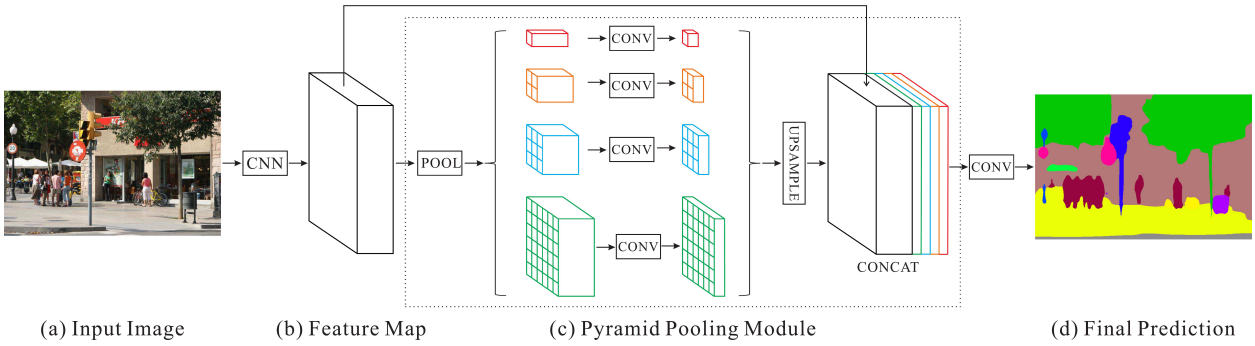


**Figure 4.16:** PSPNet Architecture [90]

layer. The structure abstracts different sub-regions by adopting pooling kernels of varying sizes in just a few steps. Therefore, multistage kernels should maintain a reasonable gap in representation. The pyramid pooling module is four-tiered with container sizes of $1 \times 1$, $2 \times 2$, $3 \times 3$, and $6 \times 6$ respectively.

### 4.1.2 Mathematical Morphology

The word morphology is formed by morpho-, from the Greek $\mu o \rho \phi \eta$ i.e. 'form' and -logy from the Greek $\lambda o \gamma \iota \alpha$ i.e. 'study, treatment', thus meaning 'study of form'. In the field of image processing, the term mathematical morphology denotes the study of the geometric structure of the image. In the mathematical context of image analysis, morphology is understood as the tool for the extraction of image components useful for the representation and description of the shape of the region and its language is set theory. Therefore, morphology is a powerful tool for numerous image processing problems. Sets in mathematical morphology represent objects in an image. For example, the set of all black pixels in a binary image is a complete morphological description of the image.

Morphology identifies neighborhood-based operators—those that relate pixels within a small local region—as highly effective tools for processing both scalar and vector images. When applied to binary images, such operators yield only two possible outcomes: 0 or 1. As a result, neighborhood operations in binary images are focused on modifying object shapes by either adding or removing pixels. As implemented for binary images, the operations are defined by Boolean algebra and so pixel combination can only be performed using logical operations [47].

#### Morphological Operations

Five main operators are defined: Dilation, Erosion, Opening, Closing, Hit-or-Miss transform. These operations, combined with different 'structuring elements', can transform an 'object' in various ways. Erosion and Dilation are the elementary operators and all the others come from these and are defined as combinations of these.

In order to understand morphological operations, it is necessary to define the Structuring Element. The image structure is 'probed' with a user-definable shape set, usually encoded by a small raster image (3×3 or 5×5). The morphological operators can be seen as an adaption of convolution to binary images where the kernel is the structuring element (SE) and the multiplication is replaced with a logical AND, and the summation with a logical OR (Boolean Operators). Therefore, the influence of morphological operations is governed by the size and shape of the mask as the structuring element.

**Dilatation and Erosion:**

Suppose having a binary image where zeros represent the foreground while ones, the background and all the values in the SE are set to 1. If at least one pixel within the neighborhood (the SE area) is a 1, indicating part of an object, the result of the operation will also be 1. Otherwise, it will be 0. This leads to dilation: the object grows, small holes or gaps are filled, and edges become smoother. Interestingly, the same effect can be achieved using a rank value filter, specifically by applying the maximum operator to binary images. If any of the pixels in the mask are 1, the maximum value is also 1—mimicking the dilation result produced by binary convolution.

On the other hand, the minimum operator has the opposite effect. The output is 1 only when the entire SE region falls within the object. This causes erosion, where thin connections are broken and small objects that don't fully contain the SE are removed entirely. Erosion can also be implemented using binary convolution with logical AND operations.

In the case of erosion, the structuring element functions analogously to a filter or mesh, where

the openings correspond to its shape. Objects that are smaller than these openings are effectively removed from the image. An object will only be retained if, at least at one location, the structuring element is fully contained within the object's boundaries; otherwise, the object or portions of it will be eliminated.

The mathematical formula for the dilation operation is shown below [35]:

$$A \oplus B = \{z|\hat{B}_z \cap A \neq \varnothing\} = \{z|\hat{B}_z \cap A \subseteq A\}$$

The expansion effect is due to the application of the structuring element near the edges. It follows from the definition that the structuring element is tilted with respect to its origin, through the operation of reflection, and shifted by z positions through a translation. The result of the operator is the set of z positions such that $B^z$ intersects at least one element of A.

The mathematical formula for the erosion operation is shown below:

$$A \ominus B = \{z|B_z \cap A \subseteq A\}$$

the following relation:

$$(A \ominus B)^c = \{z|B_z \cap A \subseteq A\}^c = \{z|\hat{B}_z \cap A \neq \varnothing\}^c = \{z|\hat{B}_z \cap A \neq \varnothing\} = A^c \oplus \hat{B}$$

**Opening and Closing:**

Building upon the fundamental operations of erosion and dilation, more advanced morphological transformations aimed at manipulating object shape for practical image processing applications are introduced.

Erosion is particularly effective for removing small, isolated objects. However, it inherently reduces the size of all remaining objects a side effect that is often undesirable. This drawback can be mitigated by applying a dilation operation immediately after erosion, using the same structuring element. The resulting combination is referred to as an opening operation.

The opening operation selectively removes objects that do not fully contain the structuring element at any location, while preserving the overall size and shape of larger objects. It is especially useful for eliminating narrow lines or features with thicknesses smaller than the diameter of the structuring element. Furthermore, it contributes to a smoothing of object boundaries.

In contrast, the dilation operation enlarges objects and fills in small holes or gaps. To reverse this expansion while preserving the filled regions, an erosion is applied after dilation, again using the same structuring element. This sequential application is termed a closing operation.

The effects of these morphological operations on object area can be succinctly characterised by the following relationships:

$$A \circ B = (A \ominus B) \oplus B$$

As regards the closing operation, this is mirrored by the opening operation and is described by the following formula:

$$A \cdot B = (A \oplus B) \ominus B$$

## 4.2 The Segmentation Pipeline

To achieve the objectives outlined in Section 2.3, the document segmentation task has been structured into two distinct phases. The first phase involves the segmentation of all major content types within a document - such as text, tables and graphs - aimed at identifying the overall layout and structural composition. The second phase focuses exclusively on the segmentation of textual components, with the goal of analysing and subdividing them into coherent semantic units, such as titles, paragraphs, abstracts, and other text-based regions.

This two-stage approach enables a progressive refinement of the segmentation process: initially, the document undergoes a broad segmentation to localise and classify all relevant objects; subsequently, a focused analysis is conducted on the textual areas to extract higher-level semantic information. The segmentation tasks are therefore executed sequentially first identifying the structural elements of the document, and then applying more detailed analysis to the text alone.

In pursuit of these goals, multiple models have been evaluated for their effectiveness in the two phases. For the first phase - the layout segmentation - two Convolutional Neural Network (CNN) architectures were tested. These end-to-end models are capable of reconstructing an input image into a segmented output, where each region corresponds to a specific document component. This allows for accurate localisation and classification of various objects contained within the document.

For the second phase - the text-only segmentation - an end-to-end multiscale CNN has been used and also a customised heuristic approach based on principles from Mathematical Morphology has been developed.

In fact, in order to be able to obtain a classification of paragraphs these must be evaluated for their homogeneity of characters which is then reflected in their size and font. These features can be captured by models that take into account variation in the scale of an object to be identified as well as by the morphology these take on.

Subsequently, another end-to-end CNN was used for the segmentation of all classes simultaneously.

The main objective, as stated in Section 2.3, is to perform both classification and localisation in order to achieve the segmentation of document components. This requirement aligns well with the framework of semantic segmentation, which allows for pixel-level classification of each document region. Such granularity is essential for distinguishing between different types of content and facilitates a structured and organized extraction of document components.

This chapter provides a detailed explanation of the methodologies and models employed for both phases, including a discussion on the rationale behind the selected approaches and their respective performance in addressing the defined objectives.

### 4.2.1 Segmentation of layout components

The implemented model aims to segment four classes: the background, all text parts indiscriminately, tables and finally graphs.

To this end, two tests were carried out using two well-known convolutional networks: the U-net [70] and the Doc UFCN network [8], both of which are explained in 4.1.1.

The former is well known for its ability to reconstruct partitioned images with high resolution, while the latter uses dilated convolution.

The exact model of the Unet implemented for the classification of layout components includes the layers as described in the paper [70] and in 4.1.1 with a modification relating to the deconvolution path. In fact, the network uses upsampling instead of the transposed convolution implemented for the project.

With regard to the Doc UFCN network, explained in 4.1.1, has a contraction path composed of five convolutional layers each of which implements a dilated convolution with a dilation rate of 1, 2, 4, 8, 16. As with the classic U-net, the number of filters is 16, 32, 64, 128 and finally 256. Each layer consists of a convolution with kernel size $3 \times 3$ dilated, followed by Dropout with a rate of 0.2, followed by another convolutional filter and finally max pooling $2 \times 2$. The expansive path, on the other hand, implements transposed convolution, as reported in 4.1.1. It therefore consists of four layers of transposed convolution, followed by the concatenation of the same layer of the decoding path, a convolution filter $3 \times 3$, dropout and another convolution filter. This path reproduces the image at the same resolution as the input and is followed by the last convolutional layer $1 \times 1$ with activation softmax and so as output a number of features map equal to 4, i.e. the number of desired classes.

The strength of this network lies in the fact that it expands the receptive field, thus considering broader features. Dilated convolution can be thought of as a zoom out of the image itself, which then captures the main characteristics of each class. In this way, it is possible to clearly identify the fundamental characteristics of the layout, thus defining the desired classes. In fact, if we consider the class of tables, for example, these differ from all other parts of the text because they are inserted into a structure composed of lines and because they are arranged in columns. If the network captured features that were too specific, it would lose the ability to define the object as a whole. In fact, the dataset evaluated on this network shows better performance than the U-net, as reported in 5.3.

### 4.2.2 Segmentation of text components

As above explained, the paragraph reports the models used for text-only segmentation. An end-to-end convolutional network was used for this purpose, and heuristics in the area of mathematical morphology were also defined.

#### CNN Architecture

In order to be able to well classify all parts of text, the end-to-end CNN MSNet, 4.1.1 capable of extracting multiscale features, was used. The architecture of the CNN used for this task, follows for the encoding path the Resnet-50 with weights pretrained for the ImageNet challenge [20, 38]. Dropout layers have been added to the Resnet-50 architecture in order to minimize the possibility of overfitting. The last convolutional layer of the resnet is the input to the MSNet module as shown in 4.1.1. That module implements in parallel $3 \times 3$, $5 \times 5$, $7 \times 7$, $9 \times 9$ convolutional layers that allow feature extraction at different levels of the receptive field preserving both local patterns and global spatial structure and permitting the correct classification of characters of different size. The decoder phase follows the Unet architecture with skip connections and Upsampling layers.

MSNet's module integrates multi-scale feature extraction and fusion into a single segmentation model tailored for high-resolution document layouts. By combining diverse receptive fields and preserving spatial detail, MSNet excels at distinguishing both large and fine-grained layout components

in a unified, end-to-end architecture.

**Heuristics in the Morphological domain:**

The idea is to make a clustering of the text into homogeneous parts. To define homogeneity, the difference between the size of the characters was chosen. In order to calculate this difference, mathematical morphology was used, which, as explained above, makes it possible to define the form of an object by means of elementary operations and the structural element with which it operates. Since morphologically a character can be represented by a cube, this was taken as the elementary reference structure. For erosion to be significant, the structuring element must be equal to the size of the average character, so the first step is to increase the resolution of the image. This is followed by dilation until an entropy value almost equal to one is reached so that the number of black pixels is almost equal to the number of white pixels. Finally, the erosion steps, always with a square elementary structure, where each connected component is clustered according to the iterations required to erode it.

- image resolution increase
- dilation with sphere elementary structure until homogeneous zones are reached:
- stop criterion: calculation of entropy if the number of black pixels is approximately equal to the number of white pixels then stop dilation
- erosion with square elementary structure: the number of erosion steps is calculated and the characters are clustered according to these

### 4.2.3 Simultaneous Segmentation of all components

In order to be able to well classify all the components of a text, a CNN with Atrous Spatial Pyramid Pooling has been implemented as explained in 4.1.1. Specifically the architecture of the CNN used follows for the encoding path the Resnet-50 with weights pretrained for the ImageNet challenge [20, 38]. Dropout layers have been added to the Resnet-50 architecture in order to minimize the possibility of overfitting. The last convolutional layer of the encoding path is the input to the Atrous Spatial Pyramid Pooling. The decoder path follows the Unet architecture with skip connections and Upsampling layers. The Atrous Spatial Pyramid Pooling (ASPP) module, which comes after the encoding path, enables the model to capture multi-scale contextual information. This is achieved by applying parallel atrous (dilated) convolutions with different dilation rates. This improves the accuracy of segmentation in documents with layouts containing objects of different dimensions.

### 4.2.4 Evaluation metrics

In semantic segmentation tasks, evaluation metrics play a critical role in assessing model performance. Since the models in use classify each pixel, their evaluation must be addressed with a statistic that determines whether they are well-classified. In fact, the output representation of the model is a map of the areas of each component, so the metrics used to evaluate the model are in function of the pixels classified [32, 69].

The definitions of the metrics used, which are derived from the confusion matrix, are given below:

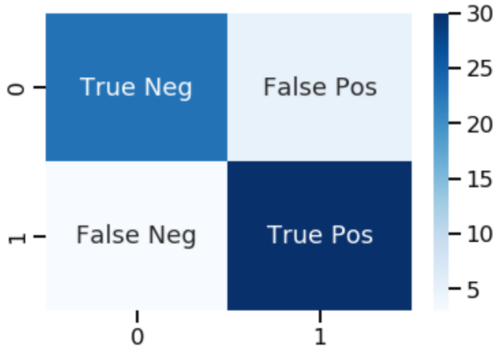


**Figure 4.17:** Confusion Matrix

- True positive (TP) represents the number of pixels correctly classified as positive
- False positive (FP) represents the number of pixels incorrectly classified as positive
- True negative (TN) represents the number of pixels correctly classified as negative
- False negative (FN) represents the number of pixels incorrectly classified as negative

**Pixel Accuracy:** is defined as the ratio of correctly classified pixels to the total number of pixels. It is calculated by the following equation:

$$\text{PA} = \frac{\sum_{i=1}^{K} n_{ii}}{\sum_{i=1}^{K} t_i}$$

where: $n_{ii}$ is the number of pixels correctly predicted for class $i$, $t_i$ is the total number of pixels belonging to class $i$, $K$ is the total number of classes.

Although PA provides a straightforward measure of segmentation accuracy, it can be misleading in class-imbalanced datasets, as it tends to favour dominant classes.

**Precision:** $\frac{TP}{TP+FP}$ measures the accuracy of the positive predictions from all the pixels classified as positive

**Recall:** $\frac{TP}{TP+FN}$ measures the accuracy of the positive predictions from all the positive samples in the dataset

**Accuracy:** $\frac{TP+TN}{TP+TN+FN+FP}$ measures the total accuracy of predictions from all the samples in the dataset

**Intersection over Union (IoU) or Jaccard index:** $\frac{TP}{TP+FN+FP}$ measures the accuracy of positive predictions from the samples not correctly classified.

For a given class $i$, the Intersection over Union (IoU) is calculated as:

$$\text{IoU}_i = \frac{n_{ii}}{t_i + \sum_{j=1}^{K} n_{ji} - n_{ii}}$$

where: $n_{ii}$ is the number of true positive pixels for class $i$, $\sum_{j=1}^{K} n_{ji}$ is the total number of pixels predicted as class $i$, $t_i$ is the number of ground truth pixels for class $i$.

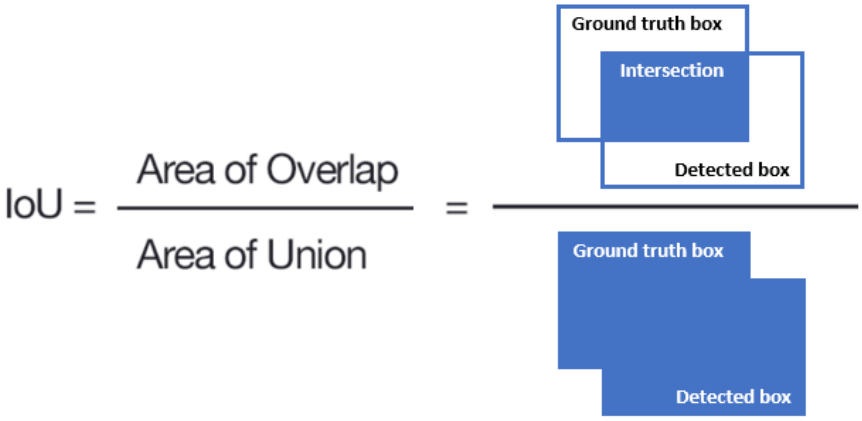


**Figure 4.18:** Intersection Over Union

It measures the overlap between a predicted class and the ground truth. Essentially, IoU determines how well the predicted semantic classes align with the real object segmentation. It computes the intersection, i.e. the area where the predicted segmentation and the actual segmentation overlap, in order to obtain the combined area of the predicted and actual segmentation. The final calculation is made by dividing the area of intersection by the area of union. IoU values range from 0 to 1, where 1 represents a perfect match (complete overlap), and 0 represents no overlap.

**Mean Intersection over Union:** [18, 28] is a more informative metric that evaluates the average overlap between the predicted segmentation and the ground truth for each class.

The mean IoU is then the average across all classes:

$$\text{mIoU} = \frac{1}{K} \sum_{i=1}^{K} \text{IoU}_i$$

mIoU is particularly valuable in scenarios with class imbalance, as it penalizes both false positives

and false negatives, thus providing a more balanced evaluation than pixel accuracy.

**DICE or $F_1$ score:** $\frac{2*TP}{2*TP+FN+FP}$ is a metric used to evaluate the performance of classification and segmentation models in cases with unbalanced data sets and is calculated as the harmonic mean of precision and recall.

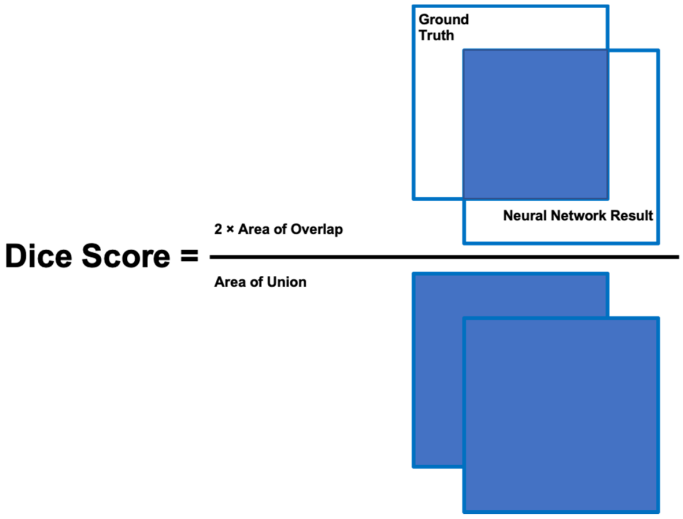


**Figure 4.19:** Dice or F1 score

# Chapter 5

# Experimental Results

This chapter begins by detailing the various datasets employed in training the models discussed in 4.2, with particular emphasis on the use of PDF documents and their transformation into image formats suitable for machine learning tasks.

Subsequently, the chapter presents a comparative evaluation of the experimental results against state of the art approaches.

The results are systematically categorised based on the type of component segmented: layout elements, text-only elements, and comprehensive segmentation involving all defined classes.

Following the performance evaluation, the chapter provides an analysis of the energy impact associated with the proposed system, including a comparison between the energy consumption of the *Sapient* system and that of human processing.

Finally, the chapter concludes with a discussion of indirect outcomes, specifically focusing on the implications for digitisation and dematerialisation.

## 5.1 Dataset

In contemporary digital workflows, virtually all types of documents are created, shared, and archived in PDF (Portable Document Format). As a result, understanding the characteristics of PDF and its transformation into image formats is crucial for the implementation of the model described in Section 4.2. This section explores the fundamental properties of the PDF format and discusses the rationale behind its conversion to raster image formats such as JPEG and PNG for processing in semantic segmentation pipelines.

The Portable Document Format (PDF) was developed by Adobe Systems in the early 1990s as a solution to the challenges of cross-platform document exchange. Its core purpose is to maintain the visual integrity of a document—including layout, fonts, graphics, and images—regardless of the device, software, or operating system used to view it [77]. The format is based on the PostScript imaging model but is more structured to facilitate interactive viewing and reliable rendering.

A PDF document may consist of one or more pages, each capable of containing a combination of text, vector graphics, raster images, and interactive elements such as hyperlinks, annotations, and form fields. Its design ensures device and resolution independence, making PDFs ideal for high-fidelity rendering across printers, screens, and platforms.

The PDF file format is composed of several well-defined components:

- Objects: These include text, fonts, images, and other resources.
- File structure: Specifies how objects are stored, accessed, and cross-referenced.
- Document structure: Defines the logical arrangement of content, such as pages, outlines, and metadata.
- Page description: Encodes the actual visual appearance of each page using graphical operators.

PDFs also support font embedding, ensuring consistent text rendering even if the viewer lacks the original fonts. In terms of performance, PDFs offer compression techniques (e.g., JPEG for images, LZW and ASCII85 for text and graphics), random access to pages via cross-reference tables, and support for incremental updates—enabling quick modifications without rewriting the entire file.

Finally, PDF is an extensible format, designed to incorporate multimedia and future content types while maintaining backward compatibility.

Given their fixed-layout nature and visual fidelity, PDFs can be accurately transformed into raster image formats such as JPEG or PNG. This conversion is essential in tasks like semantic segmentation, where pixel-level classification is required. The reasons for high-accuracy conversion include:

1. Fixed Layout: Each element in a PDF is positioned precisely, enabling an accurate visual snapshot when rasterized.
2. Device Independence: The layout is preserved across different resolutions and platforms.
3. Advanced Rendering Engines: Tools like Adobe Acrobat, PDFium, and Ghostscript can render each page with high precision at configurable resolutions (DPI).
4. Embedded Resources: Fonts and images are embedded within the file, ensuring consistency and eliminating external dependencies during rendering.

As a result, each PDF page can be treated as a visual canvas, structurally rich but ultimately represented as a two-dimensional image suitable for computer vision tasks.

Once converted, PDF pages are typically stored in JPEG or PNG formats, both of which have distinct characteristics suited to different processing needs.

The JPEG format is based on a lossy compression algorithm designed in the early 1990s, ideal for compressing complex color images such as photographs. While JPEG itself is not a file format, implementations like JFIF (JPEG File Interchange Format) and SPIFF (Still Picture Interchange File Format) serve as container formats. JPEG supports 24-bit color depth and achieves high compression ratios, making it suitable for large-scale document datasets.

JPEG's main advantages include:

- Compact file size
- Broad compatibility
- Efficient storage for scanned documents or photograph-heavy pages

However, its lossy nature may introduce artifacts in high-detail regions, making it less suitable for tasks requiring pixel-level precision.

The PNG format, introduced in 1996, provides a lossless compression alternative. It uses a variant of the LZ77 algorithm and supports color depths from 1-bit to 48-bit, transparency, interlacing, and error detection mechanisms. As an open format without licensing restrictions, PNG has become popular for applications requiring crisp, artifact-free image representation—such as text-heavy documents and diagrams.

Notable advantages of PNG include:

- Preservation of image quality
- Support for transparency (useful in document overlays)
- Rich metadata support

PNG is particularly valuable when accurate preservation of text and line-art is essential, as in document layout analysis and OCR preprocessing.

The rationale for Image Format Selection used in this project, involves the choice of image format for PDF page rendering which is influenced by the trade-off between file size, image quality, and processing requirements. PNG is preferable for lossless processing and precise text segmentation, while JPEG may be used for faster processing of visually complex pages where minor losses in detail are acceptable.

The conversion of PDF documents into these image formats enables the application of deep learning models, such as convolutional neural networks, for layout detection and semantic segmentation—thereby bridging the gap between structured document representation and image-based pattern recognition.

## 5.2 Datasets used in this project

In order to train and evaluate the classification pipeline, several datasets were used which are described here.

For the training of the layout components model described in 4.2.1 the *PubLayNet* dataset [91] has been used. It is a large-scale dataset designed for document layout analysis, specifically targeting academic publications. PubLayNet annotations are provided in COCO format and each document page is annotated with bounding boxes for layout elements. The annotated categories include:

- Text – Paragraphs or blocks of body text.
- Title – Titles of sections or the paper itself.
- List – Bulleted or numbered lists.
- Table – Tabular data.
- Figure – Graphs, charts, and images.

For the specific aims of the model described in 4.2.1, the dataset has been composed of four classes: all the text parts without distinction, tables, figures and finally white background. In order to adapt the dataset for semantic segmentation, each of the above classes was assigned a colour and each bounding box was modified so that all pixels within or on the boundary of the box were labelled with the relevant class.

The dataset comprises 2,000 images and their respective masks. All of the images are in RGB PNG format, and the grayscale PNG masks are categorised into four classes, which are represented by the array [0, 1, 2, 3].

The results of the model training with the described dataset are reported in 5.3.1

For the training of the text-only model, reported in 4.2.2, the first 79 PDFs were taken from the *PMC Sample 1943* dataset [65] a public dataset of scientific articles, comprising 1,943 articles in the Pubmed repository, and were manually annotated for a total of 808 images of publication data. Each paper in PDF has been annotated using Adobe Acrobat's redaction tool, thanks to which each component has been colour-coded with an RGB colour, so that the document is totally annotated in all its parts with a different colour. A copy of the PDF was then saved giving a redacted or masked version of the document and the original document itself. The original document is converted to an RGB PNG image, while the masked document is converted to a grayscale PNG. The final components annotated were: Title, Section Header, Authors' names, Text, Abstract, Table, Caption, Figure, Footnote, References, Page Footer and Header.

To obtain a broader classification system for text classes, the dataset was expanded to include images from sources other than scientific papers. This enabled a more comprehensive modelling of different font types and increased the size of the dataset. For this purpose, the DocLayNet dataset [64] has been used. The DocLayNet dataset is a modern and comprehensive benchmark for document layout analysis, designed to reflect the diversity and complexity of real-world documents. It was introduced to address limitations of older datasets like PubLayNet by including a broader range of document types, better-quality annotations, and a richer label taxonomy. In fact, scanned and computer native documents were taken from various domains such as: government records, scientific articles, legal contracts, magazines and books.

Annotations in DocLayNet go beyond basic bounding boxes. They are provided in COCO format, including pixel-accurate segmentation masks for all layout elements. As with Publaynet, the masks have been modified in order to apply them to the model. The dataset comprises 11 categories: Title, Section Header, Text, Table, Caption, Figure, Footnote, Formula, List, Page Footer, Page Header. From this dataset 2000 images and their relative masks have been used.

The results of the model training with the described dataset are reported in 5.3.2

## 5.3 Results obtained and comparison with Literature

The paragraph reports the results obtained and the relative models used, including a comparison with the state of the art.

All models have been trained on Google Colaboratory, commonly known as Colab, a cloud platform based on Jupyter Notebook, which allows the execution of the Python code in the browser, leveraging the hardware resources provided by Google, specifically the T4 GPU has been used.

As regards the segmentation of layout components, i.e. text, tables and images, two different models were used: Unet [70] and Doc UFCN [8], described in section 4.1.1.

For text segmentation alone, however, the heuristic described in section 4.2.2 and the CNN described in section 4.2.2 were used.

Finally, for the segmentation of all the components the model is described in 4.2.3.

The results of each model are reported alongside the dataset used for that model. Therefore, the first paragraph refers to the segmentation of layout components 4.2.1, while the second refers to the segmentation of text parts 4.2.2.

### 5.3.1 Results related to layout components

For training the first model, the *PubLayNet* dataset [91] reported in 5.2 was used, specifically 2000 images representing all text parts, tables and figures and randomly divided into training (60%), evaluation (20%) and test sets (20%) resized to $512 \times 512$ pixels. The images used were in RGB PNG format while the relative masks in grayscale PNG format. The first test was carried out using the U-net model, in which the deconvolution phase consists of a transposed convolution. The Loss function used is the categorical cross entropy and as optimizer Adam with Learning rate of $10^{-4}$. Training was set up with a batch size of 3 and 50 epochs, but thanks to early stopping, it stopped at 18, achieving an accuracy of 94% and IoU of 80%.

Below are images of the training and validation assessments.

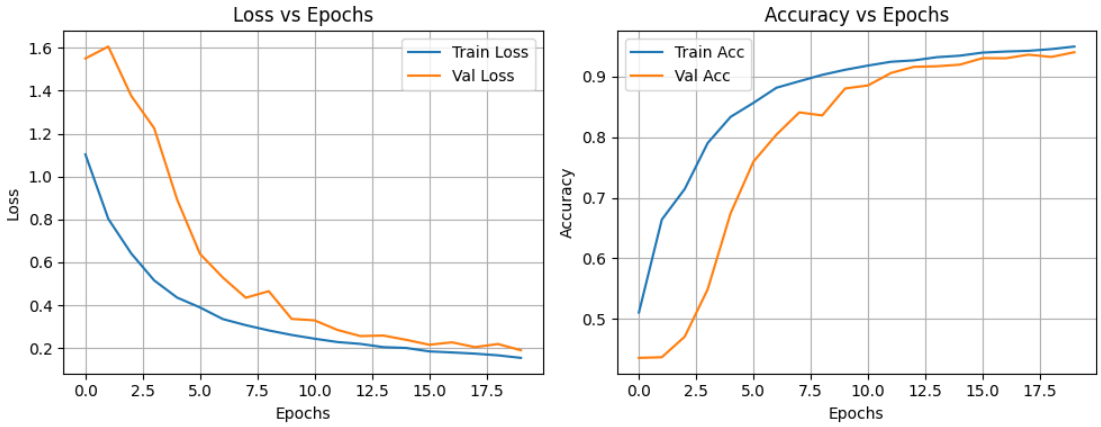


**Figure 5.1:** The evaluation of the model

The second test was carried out using the Doc UFCN architecture, reported in 4.1.1. The Loss function used is the categorical cross entropy and as optimizer Adam with Learning rate of $10^{-4}$. Training was set up with a batch size of 3 and 50 epochs, but thanks to early stopping, it stopped at 34, achieving an accuracy of 97.72% and IoU of 90.15%.

Below are images of the training and validation assessments and three examples of segmentation.

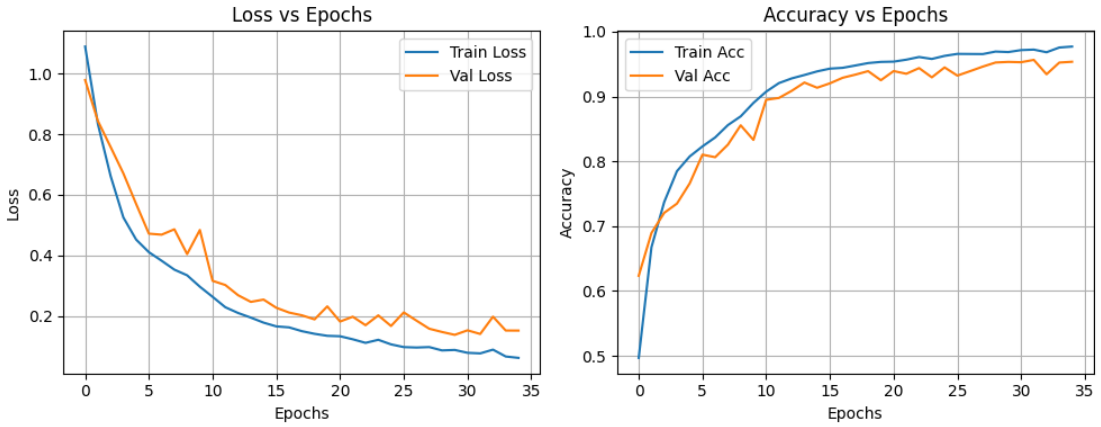


**Figure 5.2:** The evaluation of the model

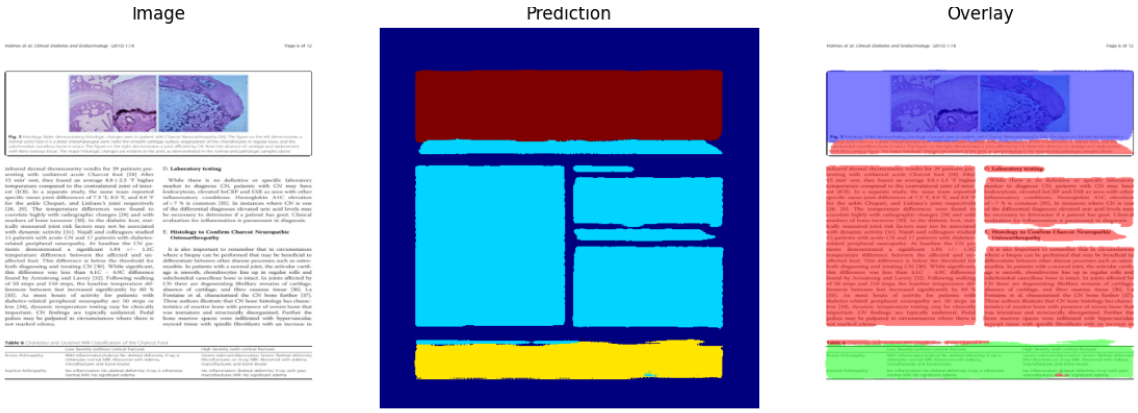


**Figure 5.3:** Overlay of segmentation

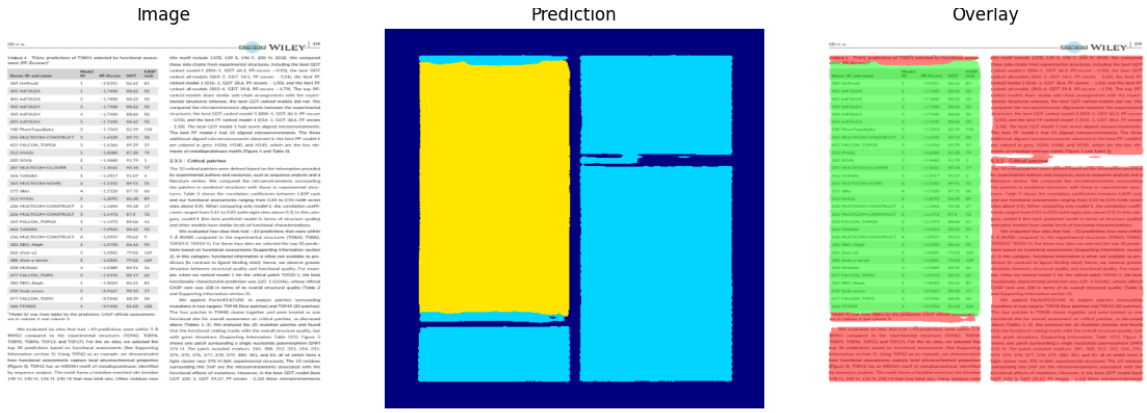


**Figure 5.4:** Overlay of segmentation

The state-of-the-art comparison is carried out objectively, thus comparing the paper [83] that determines the segmentation of the same classes, albeit with a very different network architecture, as it implements 4.1.1. The paper reports an IoU of 90.55%.

### 5.3.2 Results related to text-only segmentation

As regards the results obtained using the CNN explained in 4.2.2, the dataset used is composed of 2833 images, the 833 manually annotated images have been expanded with 2,000 RGB images of $512 \times 512$ from the DocLayNet dataset, which have also been modified so that the abstract class and authors not present in the dataset can also be obtained. So the final classes were 11 and comprise: Title, authors, abstract, header, footer, paragraph, title of paragraph, captions, notes and lists. The Loss function used is the so called Combination Loss $Loss = \alpha * ce + (1 - \alpha) * dsc$ where the scalar $\alpha$ has been set to 0,5 and $ce$ represents the categorical cross entropy while $dsc$ the Dice Loss. As optimizer Adam with Learning rate of $10^{-4}$. Training was set up with a batch size of 5 and 50 epochs, but thanks to early stopping, it stopped at 18, achieving an accuracy of 96.5% and IoU of 80.15%. The figures represents the training and validation Accuracy and Loss and two examples of predictions.

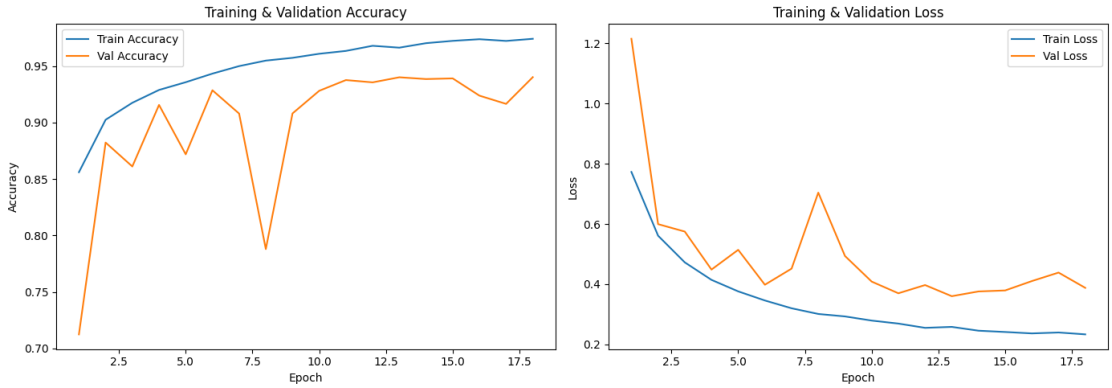


**Figure 5.5:** Training evaluation over epochs

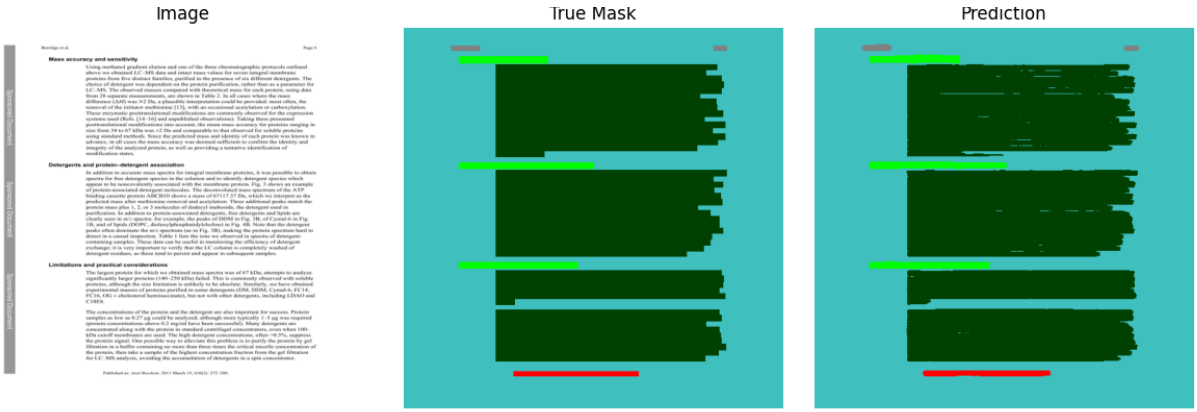


**Figure 5.6:** Prediction

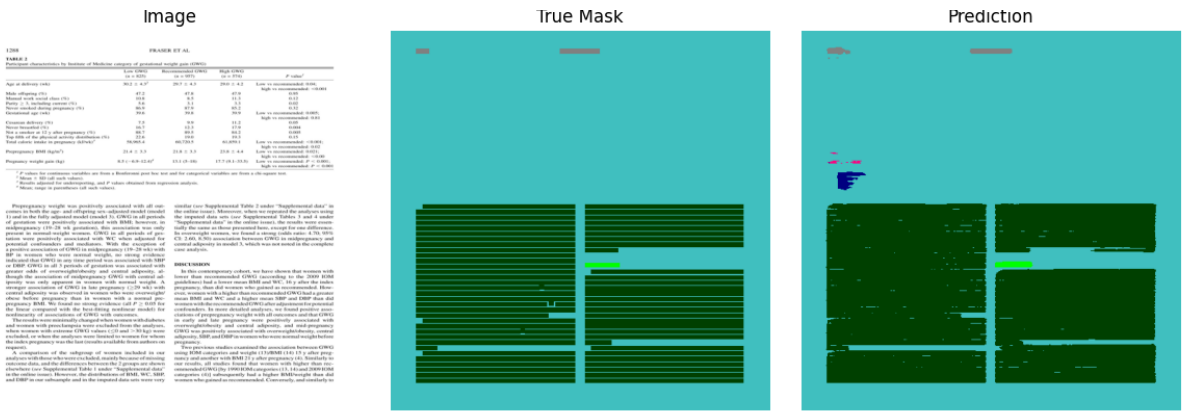


**Figure 5.7:** Prediction

As regards the results obtained using the heuristic explained in 4.2.2, an evaluation was carried out based on the average intersection over union of 50 outputs generated with the relative masks, which was reported as 99.3%.

The high result is possible thanks to the fact that the model does not require an upsampling phase, as the heuristic operates directly on the characters. The connected components of the perfectly scanned text-only images each represent a single character, and it is the number of iterations required for the erosion operation that determines their classification.


**Figure 5.8:** Output of Heuristic 4.2.2

The model could be compared with the paper Historical document analysis [66]. Although the paper investigates cases where few training samples are available, its main objective is region classification. This includes detecting dominant blocks, such as highlights, main text, glosses, drop caps, initials, and other ornaments. The paper proposes using different variants of the UNet network, achieving an IoU of 81.71%.

### 5.3.3 Results related to all components

As regards the results obtained using the CNN explained in 4.2.3, the dataset used is composed of 2833 images, the 833 manually annotated images have been expanded with 2,000 RGB images of $512 \times 512$ from the DocLayNet dataset, which have also been modified so that the abstract class and authors not present in the dataset can also be obtained. So the final classes were 13 and comprise: title, authors, abstract, header, footer, paragraph, title of paragraph, graphs, captions, tables, notes and lists. The Loss function used is the so called Combination Loss $Loss = \alpha * ce + (1 - \alpha) * dsc$ where the scalar $\alpha$ has been set to $0,5$ and $ce$ represents the categorical cross entropy while $dsc$ the Dice Loss. As optimizer Adam with Learning rate of $10^{-4}$. Training was set up with a batch size of 10 and 50 epochs, but thanks to early stopping, it stopped at 12, achieving an accuracy of 95.79% and IoU of 74.89%.

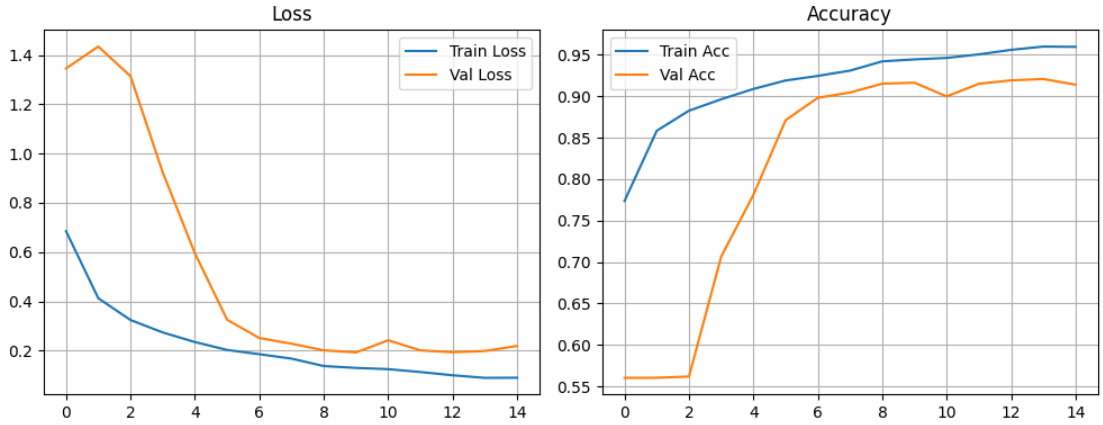


**Figure 5.9:** Training evaluation over epochs 4.2.2

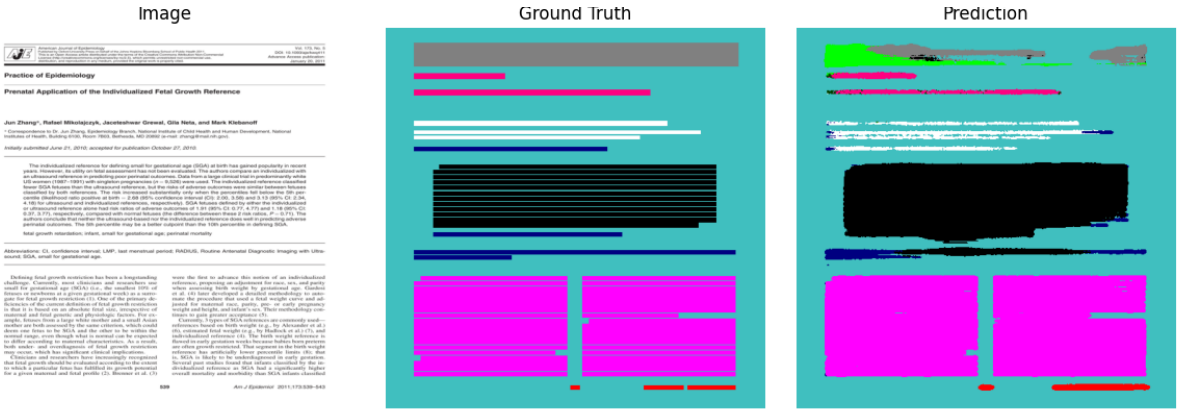


**Figure 5.10:** Prediction 4.2.2

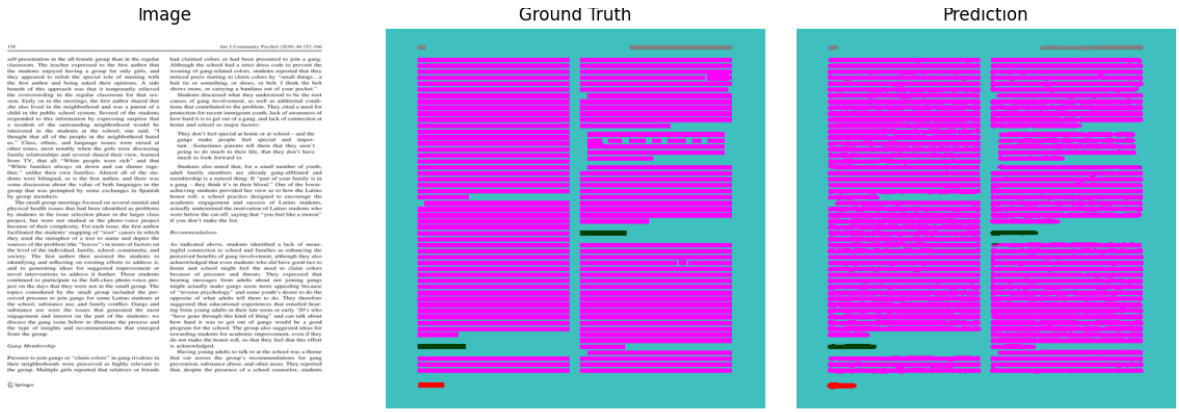


**Figure 5.11:** Prediction 4.2.2

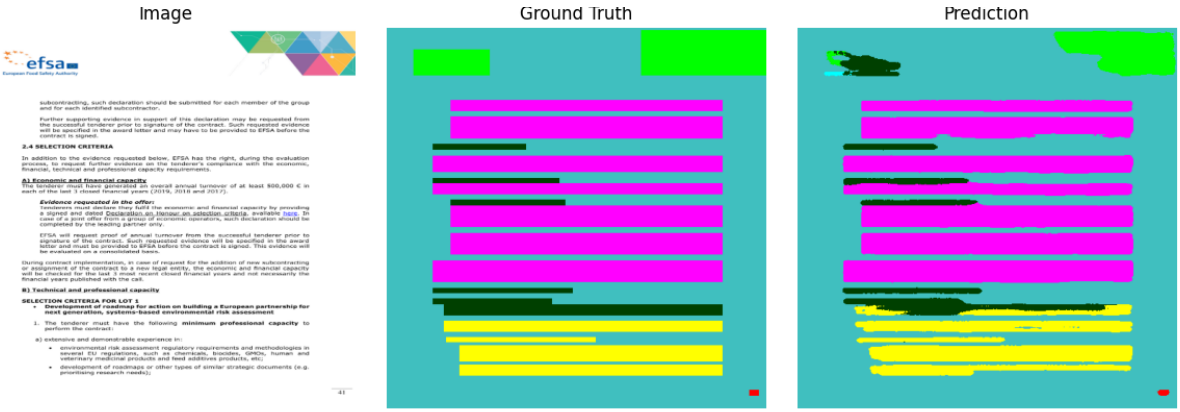


**Figure 5.12:** Prediction 4.2.2

For simultaneous segmentation of the above classes, the reference paper is [58] although this yields excellent results in many classes. The paper investigates more models and defines also an architecture that is a variant of Unet with Resnet-50, to which the pyramid pooling module is concatenated.

However, the Deeplab v3+ 4.1.1 architecture has produced the best results in 43 classes, achieving 78.08% IoU, thanks also to the definition of a bounding box regression method.

## 5.4 Improvements

Aspects relating to reducing environmental impact are taken into consideration, comparing the performance of the software produced by the project with that of a human user. The methodology for simplified assessment of the impact of the software-hardware (SW/HW) relationship of tools in use in the Company *Expert.ai* is reported below. For an approximate assessment of the environmental impact that the use of SW/HW tools for the execution of simple or complex tasks can have, it is advisable to proceed with a human-machine comparison considering three main aspects:

- Power consumption
- Emissions and resource consumption
- Time and productivity

In the first analysis, it is possible to proceed with the general quantitative evaluation of the previous points by measuring the individual items in relation to predetermined units of time (day, month, year). Secondly, these measures can be specified by referring to certain and circumscribed tasks/problems.

### 5.4.1 Comparison between enterprise application integration (EAI) systems and human energy consumption

The Proximity Placement Groups (PPGs) has 7 Virtual Machines (VMs) and 3 physical hosts on which as many servers and machines for batch development/execution are implemented, with various HW and SW configurations. To estimate power consumption with reasonable accuracy, it is sufficient to normalise this set of machines as if they were 'hosted' on a single server/rack hardware component: EAI usually equips itself with machines equal or similar to the DELL PowerEDGE R720, which can be taken as a reference to estimate a general power consumption. This HW component mounts two redundant power supplies (one of which serves as a backup) which have a nominal power (peak - therefore not continuous and not at full capacity) of 870W. To measure energy consumption, given and known the value of the potential ($V$) supplied by the electricity supplier ($220/230V$ – domestic contracts, $480V$ for company/three-phase contracts), it is necessary to use an ammeter to measure the intensity of electric current ($I$) absorbed by the device at a given instant: the product of these two values returns the value of instantaneous power absorbed by the machine ($P = VxI$); this value must be compared to the unit of time to obtain the energy consumption expressed in Wh or kWh. Since this procedure is difficult to implement, the energy consumption of the machine is estimated, which can be approximated with about half of the nominal power (approx. $435W/h$) added to a percentage of peak consumption that depends on the use of the device. This is possible if the following considerations are made: a. In steady state (without absorption peaks due to SW processes that increase the energy demand to the HW components), the rate of the power supplies can be approximated to a sine wave with a continuous component equal to half of the nominal power; b. In peak state (SW processes that consume energy), the behavior of power supplies is still approximate to a sine wave, but with a higher continuous component than in steady state, which depends on the percentage of time that the machine is in peak state compared to steady state. If we hypothesize on an empirical basis a peak regime of 30% compared to the

total power-on time, during which the device absorbs an instantaneous average power of $650W$, we can estimate the power consumption by weighted average: $(70 * P_s + 30 * P_p)/(30 + 70) = 499$ 5W Where $P_s$ state as steady-state power and $P_p$ refers to peak power. We therefore assume an energy consumption of approximate 500 watt-hour ($Wh$), or 12kWh per day, 4380kWh per year. In fact, the most modern HW components are equipped with energy-saving tools that, in stationary conditions, allow consumption to be reduced by up to 50%, and which were not considered in these estimates.

### 5.4.2 Evaluation of the energy consumption of a specific task with HW EAI tools – PPGs

Based on the above calculations, we can proceed with the calculation of the energy consumption of a single SW process running on the HW systems supplied. In particular, a running process is actually executed by only one of the VMs installed on the machine, which is why it does not take up all the 100% HW resources, which are sized to perform many tasks at the same time (managed by multiple VMs). Previously, we assumed that all ten of our VMs are hosted by our HW component, so we can estimate a consumption of 10% of the total for each (rough estimate), or 50 Wh in "mixed" mode. For example, a crawling job that takes 2 hours to execute will consume 0.1 kWh (which is the portion of energy consumed by the SW, not by the entire HW, where SW means the whole of the OS and the crawling process).

### 5.4.3 Evaluation of human energy consumption

To estimate human energy consumption, we must use the daily averages of an adult person (average estimated taking into account both rest periods and study/work periods). In particular, the daily energy consumption of an adult, on average, is indicated as follows: $2000kCal/day = 8400kJ/86400s = 2.326kW = about100Wh$ It should be emphasized that this energy consumption is not due in particular to work task activities, but is such given the normal existence of the individual. Let's now evaluate how much it costs in terms of $CO_2$ emissions to produce electricity. In Italy, the estimated $CO_2$ production is $406g/kWh$, i.e. 406 g of $CO_2$ emitted for every kWh of energy. The crawling job in the previous example consumed $0.1kWh$, which is equivalent to about $40g$ of $CO_2$, while the HW component in use by the FP group is responsible for emitting $12x406 = 4.87kg$ of $CO_2$ per day. In the case of humans, on the other hand, assessing $CO_2$ emissions is more complicated, as it would be necessary to evaluate $CO_2$ emissions for each calorie consumed. The greatest difficulty, however, is to evaluate the food source of the calories consumed by the individual, and to evaluate the $CO_2$ emission to produce every possible food. For this reason, we will proceed with an estimate. According to a German study, the environmental impact of food production (assessed per kg of $CO_2$ produced for each kg of food) varies from about one kg for starchy foods, a couple of kg for fruit and vegetables to about 14 kg for beef. For a correct evaluation, it would be necessary to estimate in what percentages these foods are consumed by an individual (also depending on the nationality) as well as the environmental impact of the food cooking process. For simplification, let's take as a model a Western/Mediterranean diet consisting of 50% starchy foods, 10% fruit and vegetables and 40% meat (white, red, fish). Assuming the consumption of 2000 Cal/day, we can estimate: $1000Cal$ from starchy foods is about $300g$ of $CO_2$ $200Cal$ from fruit and vegetables is

approximate $500g$. So $1kg$ of $CO_2$ ia about 800 Cal from meat. 3 kg of $CO_2$ total: 4.3 kg $CO_2$ per day is comparable to the emission of our HW component.

### 5.4.4 Consumption of material resources

The consumption of natural resources can be estimated on the one hand in the consumption of water for energy production and human activities, and on the other hand by the consumption of resources to perform a specific task. In our crawling job example, a human would likely consume a large amount of paper, which is saved by using the technology. As far as water consumption is concerned, this can be done in the same way as for $CO_2$ emissions.

## 5.5 Indirect results

One of the indirect aspects is the digitisation and dematerialisation of the documents processed by the systems described above. Such applications are currently a necessity for both private and public companies. It is indeed a common goal to reduce paper documents (paperless office), i.e. to convert them into digital format. One of the indirect yet increasingly critical aspects of implementing the systems described above is the digitisation and dematerialisation of the documents they process. This process involves converting physical, paper-based documents into digital formats that can be more efficiently stored, retrieved, and analysed. A central technological enabler of this transformation is optical character recognition (OCR), which allows for the automatic extraction of textual information from scanned images or photographs of documents. OCR technologies, particularly those enhanced by machine learning and deep learning algorithms, have significantly improved in accuracy and versatility, making them suitable for processing a wide variety of document types — from structured forms to unstructured handwritten notes.

Such applications are no longer optional; they are now essential for both private enterprises and public sector organisations that must manage growing volumes of information while ensuring accessibility, security, and compliance with regulatory standards. The broader aim is the creation of a paperless office, wherein paper documents are minimised or entirely replaced by digital equivalents. Beyond reducing physical storage needs and operational costs, this transition facilitates automation, enhances data availability for downstream machine learning tasks, and supports environmentally sustainable practices. By embracing digital documentation, organisations can streamline workflows, improve data accuracy, enhance security through encryption and access controls, and facilitate remote or hybrid work environments. This shift also supports broader sustainability goals by reducing the reliance on paper, printing, and physical storage. The digitisation and dematerialisation of documents thus represent not only an administrative improvement but also a strategic foundation for developing intelligent, data-driven systems in contemporary organisations.

### 5.5.1 Digitisation

Once a document has been digitised it can be handled by different kinds of systems, making increasingly agile all those processes that can be automated. In particular, the main purpose is the provision of services that can be used regardless of the place where you work and the IT tools

you use. In fact, digitisation allows the documents produced or converted to be shared and stored through IT platforms that are delocalised with respect to user access.

Digitisation therefore means abandoning traditional, so-called analog tools, in order to streamline workflows and automate activities and procedures, whether related to the production and distribution of products and services, marketing and sales, customer care, or pertinent to the document management of administrative, accounting and tax material. Digitisation is therefore a necessary choice, capable of bringing many advantages, first of all, the elimination of paper is a sustainable choice and allows you to save space: everything is stored and managed through a computer without archives and cabinets. Secondly and finally, the acquisition of new digital tools has significantly reduced the time required to access a particular document or complete a task.

In the case of the Public Administration, digital innovation is a necessity for the efficiency of the services provided but also because it allows to seize all opportunities both within the administrations and in the relations between them with citizens and businesses. The objectives it pursues are related to the promotion of digital citizenship rights and forms of participation favoured by a government that promotes the development of public platforms for the provision of services, the expansion of digital citizenship and the sharing of ideas and information. In addition, the aim is to obtain transparency in order to be able to access all the information necessary for the functioning and work of public administrations.

### 5.5.2 Dematerialisation

Dematerialisation is the digitisation process that aims to initiate processes of digital translation of paper documents necessary to carry out activities. Once dematerialised, therefore, the documents can also be used online, and this allows companies and Public Administrations that have procedures and tools for carrying out work even remotely to provide all services and products as in physical spaces.

This process involves a series of operational and technological adaptations that are essential for everything to work in a coherent and safe way. Dematerialisation does not simply mean eliminating paper document supports and the physical archives that contain them, but contextualizing them in the broader context of digitisation. This is only the first, essential step that must be taken in the field of document management. This means that infrastructures, practices and solutions must be built around dematerialisation initiatives in the strict sense that allow users and machines to access, use and store documents in their new format.

The term dematerialisation therefore indicates the progressive increase in computerised document management within an organisation, with the gradual replacement of traditional documentation supports with adequate information systems. The main objective of dematerialisation processes is to convert paper documents into electronic documents. It is necessary to preserve, thanks to appropriate certified platforms, including electronic signature platforms, both their legal value, and the elements that allow their identification in the company and reference archival context. The conversion of paper documents into electronic documents is the basis of the preservation that intends to replace traditional folders and archives with databases and allows documents to be stored in a more secure and environmentally friendly way and which can be made available to anyone who has the credentials to view and modify them. In the same way, it becomes possible to eliminate the original paper documents of which an electronic copy has been produced. The primary objective of

the gradual elimination of paper, through the computerisation of processes, is to simplify relations between the Public Administration and citizens or businesses. It is however clear that there would also be advantages in terms of the efficiency of document management, the quality of the work of public employees and the containment of operating costs.

# Conclusions

This Thesis has presented an in-depth investigation into the problem of Document Layout Analysis using Convolutional Neural Networks (CNNs), with a particular emphasis on the segmentation of components in text-based images. The overarching objective was to develop a method capable of performing segmentation at two complementary levels: the *layout level*, which involves the identification and differentiation of structural regions within a document (e.g., text sections, images, tables, and graphic elements), and the *text level*, focusing on the finer-grained segmentation of textual elements (e.g., Title, paragraphs, notes etc.). These steps are critical in the broader context of Optical Character Recognition (OCR) systems, where precise segmentation directly influences the quality and reliability of subsequent character recognition and text extraction processes.

To address this problem, a multi-phase model was designed. Initially, the segmentation task was approached in two distinct stages to separately tackle layout segmentation and text-level segmentation. This phase design facilitated targeted optimization for each problem space and allowed for the exploration of specialised architectures and techniques suited to each task. Later in the project, insights from both phases were integrated into a unified framework - a single end-to-end Convolutional Network capable of learning all relevant document regions simultaneously. This unification resulted in improved coherence, reduced system complexity, and streamlined training and inference workflows.

The core architectural paradigm employed throughout this work was that of Semantic Segmentation, implemented through Fully Convolutional Networks (FCNs). These networks, by replacing traditional fully connected layers with convolutional layers, enabled spatially invariant processing of images, preserving location information essential for segmentation tasks. FCNs were particularly suited for learning both coarse and fine-grained representations, making them ideal for document image analysis. The encoder-decoder structure allowed the model to extract hierarchical features in the encoding phase while reconstructing pixel-wise segmentations in the decoding phase, ensuring both semantic understanding and spatial precision.

For layout segmentation, a CNN leveraging *dilated convolutional layers* was developed. This approach allowed the model to capture large receptive fields and aggregate contextual information from a broader area without increasing the number of parameters or computational cost. As a result, the network demonstrated strong performance in identifying and classifying various document regions including text sections, background, graphical illustrations and tabular structures.

The text-level segmentation task introduced unique challenges due to the variability in font styles, character spacing, and writing orientations across documents. To tackle these, two complementary approaches were adopted. First, a *multiscale end-to-end CNN* was implemented to enable robust feature extraction across multiple resolutions. This architecture was particularly effective

in capturing the nuanced features of different characters and text patterns, facilitating accurate segmentation at various scales.

Second, a *heuristic method based on Mathematical Morphology* was introduced. By the employing of the morphological operation of the erosion, the heuristic method was able to segment text regions based on geometric and structural cues. Although more rule-based and less flexible than learning-based methods, this approach provided valuable complementary insights and demonstrated practical effectiveness.

To unify both layout and text segmentation under a common framework, the final phase of the project implemented an end-to-end convolutional network based on the *DeepLab* architecture. Known for its use of Atrous Spatial Pyramid Pooling (ASPP) and deep feature extraction, the model offered a high degree of accuracy and adaptability across document types. This architecture enabled simultaneous segmentation of all relevant document components, making it a robust and scalable solution.

Beyond the technical contributions, this research advances the field of document image analysis by proposing a systematic and modular approach to segmentation. It provides a viable alternative to traditional rule-based or shallow learning techniques, many of which lack the flexibility or precision required for modern, diverse document formats. The use of deep learning -particularly CNN-based architectures- demonstrated the ability to generalize across varied layouts and character structures, a key requirement for real-world applications such as digital archiving, automated document indexing, and intelligent content retrieval.

The outcomes of this research open up several promising directions for future work. Further improvements could be achieved by incorporating attention mechanisms or transformer-based architectures to better capture long-range dependencies within documents. Moreover, extending the segmentation approach to handle non Manhattan layout or handwritten text could significantly broaden its applicability.

In conclusion, the models and methodologies developed in this Thesis offer a comprehensive and flexible solution to document segmentation challenges. The research not only contributes to the academic understanding of semantic segmentation in the context of document images but also provides a strong foundation for future innovations in OCR and automated document understanding systems. Through the use of deep learning, particularly Convolutional Neural Networks, this work demonstrates the potential for achieving high-precision, scalable, and adaptable document analysis solutions, paving the way toward more intelligent and autonomous systems for digital document processing.

# Bibliography


[1] Agenzia per l'italia digitale. https://www.agid.gov.it/en/agenzia/stampa-e-comunicazione/notizie/2024/02/13/published-2024-2026-three-year-plan-information

[2] Agenzia per l'italia digitale. https://www.agid.gov.it/it/strategia-cloud-pa.

[3] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-dujaili, Y. Duan, O. Al-Shamma, J. I. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8, 2021.

[4] S. Ares Oliveira, B. Seguin, and F. Kaplan. dhsegment: A generic deep-learning approach for document segmentation. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, page 7–12. IEEE, Aug. 2018.

[5] V. Badrinarayanan, A. Handa, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling, 2015.

[6] Y. Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, Jan. 2009.

[7] Y. Bengio. *Learning Deep Architectures for AI*. Now Publishers, 2009.

[8] M. Boillet, C. Kermorvant, and T. Paquet. Multiple document datasets pre-training improves text line detection with deep neural networks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, page 2134–2141. IEEE, Jan. 2021.

[9] R. Cattoni, T. Coianiz, C. M. Modena, and S. Messelodi. Geometric layout analysis techniques for document image understanding: a review. *Technical Report 9703-09, IRST, Trento, Italy*, 1998.

[10] I. Change. Ipcc. *climate change*, 2014.

[11] L. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017.

[12] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2016.

[13] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2016.

[14] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017.

[15] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation, 2017.

[16] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

[17] J. Cheng, H. Li, D. Li, S. Hua, and V. Sheng. Survey on image semantic segmentation using deep learning techniques. *Computers, Materials & Continua, Tech Science Press*, 2022.

[18] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[19] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, Dec. 1989.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[21] M. Diem and R. Sablatnig. Recognizing degraded handwritten characters. *CVL Technical Report*, 07 2010.

[22] V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning, 2018.

[23] R. H. Enns. *It's a Nonlinear World*. Springer Undergraduate Texts in Mathematics and Technology (SUMAT), 2011.

[24] European commission, public administration reform under the national recovery and resilience plan. `https://commission.europa.eu/business-economy-euro/economic-recovery/recovery-and-resilience-facility/country-pages/italys-recovery-and-resilience-plan_en`. European Commission, Public Administration Reform under the National Recovery and Resilience Plan.

[25] `https://eur-lex.europa.eu/EN/legal-content/summary/e-cohesion-electronic-and-paperless-ad` `html`. European Commission, e-Cohesion – electronic and paperless administration, 2014.

[26] `https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation`. European Commission, eIDAS Regulation on electronic identification and trust services, 2014.

[27] European commission, einvoicing regulation, 2014. `https://ec.europa.eu/digital-building-blocks/sites/display/DIGITAL/eInvoicing`. European Commission, eInvoicing Regulation on electronic invoicing in Europe.

[28] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[29] J. Feng, X. He, Q. Teng, C. Ren, H. Chen, and Y. Li. Reconstruction of porous media from extremely limited information using conditional generative adversarial networks. *Physical Review E*, 100, 09 2019.

[30] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[31] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez. A review on deep learning techniques applied to semantic segmentation, 2017.

[32] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. G. Rodríguez. A review on deep learning techniques applied to semantic segmentation. *CoRR*, abs/1704.06857, 2017.

[33] H. Gholamalinezhad and H. Khosravi. Pooling methods in deep neural networks, a review. *CoRR*, abs/2009.07485, 2020.

[34] B. G.M. and M. S.A. Document layout analysis: A comprehensive survey. *ACM Comput. Surv. 52, 6, Article 109*, 2019.

[35] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., USA, 2006.

[36] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. Book in preparation for MIT Press.

[37] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 06 2014.

[38] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[39] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[40] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *arXiv preprint arXiv:1704.04861*, 2017.

[41] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7132–7141, 2018.

[42] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal of Neurophysiology*, 28(2):229–289, 1965. PMID: 14283058.

[43] D. Hunter, J. Salzman, and D. Zaelke. Glasgow climate summit: Cop26. *UCLA School of Law, Public Law Research Paper No. 22-02*, 2021.

[44] IBM. 1287 optical reader.

[45] International Conference on Document Analysis and Recognition, 2015, https://iapr.org/archives/icdar2015/index.html

[46] Italian government (legislative decree no. 82/2005). `https://www.agid.gov.it/sites/default/files/repository_files/leggi_decreti_direttive/dl-7-marzo-2005-82_0.pdf`.

[47] B. Jähne. *Digital Image Processing 6th Edition*. Springer, Berlin [u.a.], 2005.

[48] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. *Computer Vision and Pattern Recognition*, 2017.

[49] B. Kong, S.Chen, and R. M. Haralick. Automatic line detection in document images using recursive morphological transforms. *Proceedings of SPIE - The International Society for Optical Engineering*, 1995.

[50] Z. Kuang, H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, and D. Lin. MMOCR: A comprehensive toolbox for text detection, recognition and understanding. *CoRR*, abs/2108.06543, 2021.

[51] Y. LeCun and Y. Bengio. *Convolutional networks for images, speech, and time series*, page 255–258. MIT Press, Cambridge, MA, USA, 1998.

[52] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989.

[53] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[54] J. Lee, H. Hayashi, W. Ohyama, and S. Uchida. Page segmentation using a convolutional neural network with trainable co-occurrence features. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1023–1028, 2019.

[55] H. W. Lin, M. Tegmark, and D. Rolnick. Why does deep and cheap learning work so well? *arXiv:1608.08225*, 2017.

[56] R. Liu, S. Yu, F. Yang, Y. Pan, and Y. Zeng. A connected components based layout analysis approach for educational documents. *The 16th International Conference on Computer Science & Education - ICCSE*, 2021.

[57] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *arXiv:1411.4038*, 2015.

[58] L. Markewich, H. Zhang, Y. Xing, N. Lambert-Shirzad, Z. Jiang, R. K.-W. Lee, Z. Li, and S.-B. Ko. Segmentation for document layout analysis: not dead yet. *Int. J. Doc. Anal. Recognit.*, 25(2):67–77, June 2022.

[59] O. Mechi, M. Mehri, R. Ingold, and N. ESSOUKRI BEN AMARA. Text line segmentation in historical document images using an adaptive u-net architecture. pages 369–374, 09 2019.

[60] Ministero dell'università e ricerca. https://www.ponricerca.gov.it/. Ministero dell'Università e Ricerca.

[61] U. Nations. Amendment to the montreal protocol on substances that deplete the ozone layer. 2016.

[62] L. O'Gorman. The document spectrum for page layout analysis. *IEEE TRANSACTIONS ON PA'ITERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 15, NO. 11*, 1993.

[63] S. Paliwal, V. D, R. Rahul, M. Sharma, and L. Vig. Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. *CoRR*, abs/2001.01469, 2020.

[64] B. Pfitzmann, C. Auer, M. Dolfi, A. S. Nassar, and P. Staar. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 3743–3751. ACM, Aug. 2022.

[65] PMC, PubMed Central, https://pmc.ncbi.nlm.nih.gov/.

[66] N. Rahal, L. Vögtlin, and R. Ingold. Historical document image analysis using controlled data for pre-training. *International Journal on Document Analysis and Recognition (IJDAR)*, 26:1–14, 05 2023.

[67] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016.

[68] V. Rezanezhad, K. Baierer, M. Gerber, K. Labusch, and C. Neudecker. Document layout analysis with deep learning and heuristics. In *Proceedings of the 7th International Workshop on Historical Document Imaging and Processing*, HIP '23, page 73–78, New York, NY, USA, 2023. Association for Computing Machinery.

[69] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[70] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[71] S. Scardapane. *Alice's Adventures in a differentiable wonderland.* self-published, 2024.

[72] S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. pages 1162–1167, 11 2017.

[73] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[74] S. Singh. Optical character recognition techniques: A survey. *International Journal of Advanced Research in Computer Engineering and Technology (IJARCET)*, 2013.

[75] R. Smith. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633, 2007.

[76] C. G. Stahl, S. Young, D. Herrmannova, R. M. Patton, and J. C. Wells. Deeppdf: A deep learning approach to analyzing pdfs. 2018.

[77] R. C. Tim Bienz. *Portable Document Format Reference Manual*. Reading, Mass. : Addison-Wesley Pub. Co, 1993.

[78] H. T. Tran, N. Q. Nguyen, T. A. Tran, X. T. Mai, and Q. T. Nguyen. A deep learning-based system for document layout analysis. In *Proceedings of the 2022 6th International Conference on Machine Learning and Soft Computing*, ICMLSC '22, page 20–25, New York, NY, USA, 2022. Association for Computing Machinery.

[79] UNFCCC. The paris agreement. In *Paris Climate Change Conference - November 2015*, 2018.

[80] *US Patent 2,026,329 1935, Reading Machine.*

[81] V. Vanchurin. The world as a neural network. *Entropy 2020, 22, 1210*, 2020.

[82] D. Vedhaviyassh, R. Sudhan, G. Saranya, M. Safa, and D. Arun. Comparative analysis of easyocr and tesseractocr for automatic license plate recognition using deep learning algorithm. In *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, pages 966–971, 2022.

[83] B. Wang, J. Zhou, and B. Zhang. Msnet: A multi-scale segmentation network for documents layout analysis. In C. Pang, Y. Gao, G. Chen, E. Popescu, L. Chen, T. Hao, B. Zhang, S. M. B. Navarro, and Q. Li, editors, *Learning Technologies and Systems*, pages 225–235, Cham, 2021. Springer International Publishing.

[84] M. Watanabe. *Pursuing the Shadows of Consciousness*, pages 27–60. Springer International Publishing, Cham, 2022.

[85] S. Wilde, A. Arenas, Y. Yari, and G. Meza-Lovon. A document layout analysis method based on morphological operators and connected components. *Latin American Computing Conference / Conferencia Latinoamericana En Informatica*, 2018.

[86] K. Y. Wong, R. G. Casey, , and F. M. Wahl. Document analysis system. *IBM journal of research and development, 26(6):647656*, 1982.

[87] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1492–1500, 2017.

[88] J. Yao and L. Huang. Complex document layout segmentation based on an encoder-decoder architecture. *Journal of Physics: Conference Series*, 2010(1):012024, sep 2021.

[89] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions, 2016.

[90] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017.

[91] X. Zhong, J. Tang, and A. J. Yepes. Publaynet: largest dataset ever for document layout analysis, 2019.

## Acknowledgements

My first thanks go to the Ministry of Universities and Research, which with the Pon Research and Innovation programme made it possible to start the project, I would also like to thank the company *Expert.ai* as co-funder of the industrial PhD.

I would like to extend my thanks to the entire Department of Applied Basic Engineering Sciences (SBAI) at *La Sapienza University of Rome*, including the Professors with whom I collaborated.

On this occasion, I would like to thank Anil Kokaram and his entire research team at Trinity College Dublin for giving me the opportunity to work in a different context.

I would like to extend my sincere thanks to my colleagues in the SBAI Department, both to those who have left and to the newcomers.

I would like to express my special thanks to my family, especially my Dad and my brother Marco, who encouraged and supported me throughout this period. I would also like to thank Christian for being the main motivating factor in my university choices.

Special thanks go to Sandra for her endless patience, her support during these long months and for making me feel like one of her daughters.

I would like to thank Arianna for the journey we have undertaken together.

In addition, my heartfelt thanks go to all the people who are close to me even in the saddest moments, always willing to support me and love me despite everything: Giulia, Carolina, Federico, Massimo, Fulvio, Prisca, Valeria, Marta, Barbara and Luisiana.

Thanks to my diving friends and especially to my 'Little Buddy' Alice.

I would like to thank Renato for the joy he has given me, and for always being there.

Finally, I would like to remember Alessia for the happy times we spent together and the will, despite everything, to achieve the goals we set ourselves.

Eleonora, Rome, Giugno 2025