Layout Analysis for Handwritten Documents

A Probabilistic Machine Learning Approach

Phd Thesis

Lorenzo Quirós Díaz

Supervised by Prof. Enrique Vidal

November 2021

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Work partially supported by the Universitat Politècnica de València under grant FPI-II/900

Abstract

Document Layout Analysis, applied to handwritten documents, aims to automatically obtain the intrinsic structure of a document. Its development as a research field spans from the character segmentation systems developed in the early 1960s to the complex systems designed nowadays, where the goal is to analyze high-level structures (lines of text, paragraphs, tables, etc) and the relationship between them.

This thesis first defines the goal of Document Layout Analysis from a probabilistic perspective. Then, the complexity of the problem is reduced, to be handled by modern computing resources, into a set of well-known complementary subproblems. More precisely, three of the main subproblems of Document Layout Analysis are addressed following a probabilistic formulation, namely Baseline Detection, Region Segmentation and Reading Order Determination.

One of the main contributions of this thesis is the formalization of Baseline Detection and Region Segmentation problems under a probabilistic framework, where both problems can be handled separately or in an integrated way by the proposed models. The latter approach is proven to be very useful to handle large document collections under restricted computing resources.

Later, the Reading Order Determination subproblem is addressed. It is one of the most important, yet underestimated, subproblem of Document Layout Analysis, since it is the bridge that allows us to convert the data extracted from Automatic Text Recognition systems into useful information. Therefore, Reading Order Determination is addressed and formalized as a pairwise probabilistic sorting problem. Moreover, we propose two different decoding algorithms that reduce the computational complexity of the problem.

Furthermore, different statistical models are used to represent the probability distribution over the structure of the documents. These models, based on Artificial Neural Networks (from a simple Multilayer Perceptron to complex Convolutional and Region Proposal Networks), are estimated from training data using supervised Machine Learning algorithms.

Finally, all the contributions are experimentally evaluated, not only on standard academic benchmarks but also in collections of thousands of images. We consider handwritten text documents and handwritten musical documents as they represent

the majority of documents in libraries and archives. The results show that the proposed methods are very accurate and versatile in a very wide range of handwritten documents.

Resumen

El Análisis de la Estructura de Documentos (*Document Layout Analysis*), aplicado a documentos manuscritos, tiene como objetivo obtener automáticamente la estructura intrínseca de dichos documentos. Su desarrollo como campo de investigación se extiende desde los sistemas de segmentación de caracteres desarrollados a principios de la década de 1960 hasta los sistemas complejos desarrollados en la actualidad, donde el objetivo es analizar estructuras de alto nivel (líneas de texto, párrafos, tablas, etc.) y la relación que existe entre ellas.

Esta tesis, en primer lugar, define el objetivo del Análisis de la Estructura de Documentos desde una perspectiva probabilística. A continuación, la complejidad del problema se reduce a un conjunto de subproblemas complementarios bien conocidos, de manera que pueda ser gestionado por medio de recursos informáticos modernos. Concretamente se abordan tres de los principales problemas del Análisis de la Estructura de Documentos siguiendo una formulación probabilística. Específicamente se aborda la Detección de Línea Base (*Baseline Detection*), la Segmentación de Regiones (*Region Segmentation*) y la Determinación del Orden de Lectura (*Reading Order Determination*).

Uno de los principales aportes de esta tesis es la formalización de los problemas de Detección de Línea Base y Segmentación de Regiones bajo un marco probabilístico, donde ambos problemas pueden ser abordados por separado o de forma integrada por los modelos propuestos. Este último enfoque ha demostrado ser muy útil para procesar grandes colecciones de documentos con recursos informáticos limitados.

Posteriormente se aborda el subproblema de la Determinación del Orden de Lectura, que es uno de los subproblemas más importantes, aunque subestimados, del Análisis de la Estructura de Documentos, ya que es el nexo que permite convertir los datos extraídos de los sistemas de Reconocimiento Automático de Texto (*Automatic Text Recognition Systems*) en información útil. Por lo tanto, en esta tesis abordamos y formalizamos la Determinación del Orden de Lectura como un problema de clasificación probabilística por pares. Además, se proponen dos diferentes algoritmos de decodificación que reducen la complejidad computacional del problema.

Por otra parte, se utilizan diferentes modelos estadísticos para representar la distribución de probabilidad sobre la estructura de los documentos. Estos modelos, basados en Redes Neuronales Artificiales (desde un simple Perceptrón Multicapa

hasta complejas Redes Convolucionales y Redes de Propuesta de Regiones), se estiman a partir de datos de entrenamiento utilizando algoritmos de aprendizaje automático supervisados.

Finalmente, todas las contribuciones se evalúan experimentalmente, no solo en referencias académicas estándar, sino también en colecciones de miles de imágenes. Se han considerado documentos de texto manuscritos y documentos musicales manuscritos, ya que en conjunto representan la mayoría de los documentos presentes en bibliotecas y archivos. Los resultados muestran que los métodos propuestos son muy precisos y versátiles en una amplia gama de documentos manuscritos.

Resum

L'Anàlisi de l'Estructura de Documents (*Document Layout Analysis*), aplicada a documents manuscrits, pretén automatitzar l'obtenció de l'estructura intrínseca d'un document. El seu desenvolupament com a camp d'investigació comprén des dels sistemes de segmentació de caràcters creats al principi dels anys 60 fins als complexos sistemes de hui dia que busquen analitzar estructures d'alt nivell (línies de text, paràgrafs, taules, etc) i les relacions entre elles.

Aquesta tesi busca, primer de tot, definir el propòsit de l'anàlisi de l'estructura de documents des d'una perspectiva probabilística. Llavors, una vegada reduïda la complexitat del problema, es processa utilitzant recursos computacionals moderns, per a dividir-ho en un conjunt de subproblemes complementaris més coneguts. Concretament, tres dels principals subproblemes de l'Anàlisi de l'Estructura de Documents s'adrecen seguint una formulació probabilística: Detecció de la Línia Base (*Baseline Detection*), Segmentació de Regions (*Region Segmentation*) i Determinació de l'Ordre de Lectura (*Reading Order Determination*).

Una de les principals contribucions d'aquesta tesi és la formalització dels problemes de la Detecció de les Línies Base i dels de Segmentació de Regions en un entorn probabilístic, sent els dos problemes tractats per separat o integrats en conjunt pels models proposats. Aquesta última aproximació ha demostrat ser de molta utilitat per a la gestió de grans col·leccions de documents amb uns recursos computacionals limitats.

Posteriorment s'ha adreçat el subproblema de la Determinació de l'Ordre de Lectura, sent un dels subproblemes més importants de l'Anàlisi d'Estructures de Documents, encara així subestimat, perquè és el nexe que permet transformar en informació d'utilitat l'extracció de dades dels sistemes de reconeixement automàtic de text. És per això que el fet de determinar l'ordre de lectura s'adreça i formalitza com un problema d'ordenació probabilística per parells. A més, es proposen dos algoritmes descodificadors diferents per reduir la complexitat computacional del problema.

Per altra banda s'utilitzen diferents models estadístics per representar la distribució probabilística sobre l'estructura dels documents. Aquests models, basats en xarxes neuronals artificials (des d'un simple perceptron multicapa fins a complexes xarxes convolucionals i de propostes de regió), s'estimen a partir de dades

d'entrenament mitjançant algoritmes d'aprenentatge automàtic supervisats.

Finalment, totes les contribucions s'avaluen experimentalment, no només en referents acadèmics estàndard, sinó també en col·leccions de milers d'imatges. S'han considerat documents de text manuscrit i documents musicals manuscrits, ja que representen la majoria de documents presents a biblioteques i arxius. Els resultats mostren que els mètodes proposats són molt precisos i versàtils en una àmplia gamma de documents manuscrits.

Agradecimientos (Acknowledgements)

En estos momentos, al mirar hacia atrás y recordar cuando decidí llevar a cabo este doctorado, me llena de ilusión recordar como tantas personas han sido parte de este proceso. Es gracias a estas personas, a su soporte, compañía y tolerancia que hoy estoy escribiendo estas líneas.

Primeramente me gustaría agradecer a una de las personas que, sin duda alguna, más impacto ha tenido en el contenido de este documento, mi director de tesis Enrique Vidal. Gracias a él he conocido el mundo del análisis de documentos manuscritos, tanto sus complejidades técnicas como la pasión por descubrir el contenido que atesoran dichos documentos. Gracias por todas las horas dedicadas, por las discusiones e ideas, y muy especialmente por su disposición a aclarar mis dudas y sus instructivos comentarios.

I will be always grateful to the reviewers for the time they spend reviewing and evaluating this work. Their enlightening comments have greatly improved many aspects of this thesis.

Gracias a mis compañeros del PRHLT por acogerme en el grupo de investigación y hacer este doctorado posible. Ha sido muy importante para mí tener vuestro apoyo en las diferentes etapas del doctorado, tanto por los conocimientos compartidos y conversaciones interesantes, como por las risas que siempre estaban presentes (aunque fuese por videoconferencia). He de mencionar especialmente a Alejandro, por su paciencia y disposición para explicarme los detalles más elusivos del mundo del HTR, y a Moisés y Vicent, por sus invaluables conocimientos y visión sobre DLA. También quiero agradecer a mis compañeros de la CPI, particularmente a Jose R. y Miguel que han estado conmigo durante las últimas etapas de mi investigación.

Finalmente, me alegra mucho poder agradecer a mi familia y amigos, tanto en Costa Rica como en España. Gracias a su incondicional apoyo durante todos estos años.

A María, por tenerme paciencia y escuchar mis historias y disparates, por soportarme en esos momentos de agobio cuando imprimir la tesis y prenderle fuego sonaba como una gran idea y, ante todo, por siempre alegrarme el día.

A mis padres y hermanos que, aunque estamos lejos físicamente, siempre han estado ahí para ayudarme. Gracias por apoyarme siempre y por animarme a seguir adelante. Sin ellos este trabajo nunca hubiese llegado a buen puerto.

Pura Vida!

Preface

The main goal of this thesis is to provide theoretically sound, efficient, practical and robust models and algorithms to obtain parts of the intrinsic structure of a handwritten document with an unknown layout. Those parts of the document layout will always be defined keeping in mind that they will be used by some Automatic Recognition System will use them to obtain as much information as possible from the documents.

In order to explain and evaluate the methods proposed, the thesis has been organized into eight chapters and two appendices. We encourage the reader to follow this document in sequential order. Nevertheless, some chapters can be skipped or read in a different order, depending on the singular interest of the reader. In that case, we believe the diagram of dependencies between chapters shown in Figure 1 will help the reader to navigate this thesis.

In Chapter 1, we introduce the problem and motivation of Document Layout Analysis on handwritten documents. Then, in Chapter 2, we provide an overview of the theoretical background on which we base our proposed methods and algorithms. This chapter can be skipped if the reader is familiar with the subject.

Chapter 3 provides an overview of the history, main definitions, and common terms related to Document Layout Analysis. This chapter should not be skipped in order to ensure a common vocabulary and understanding of the topic.

Chapter 4 covers the definition and formalization of the proposed probabilistic methods that aim to address the Baseline Detection problem. Also, we provide an overview of the Text Line Segmentation and Text Line Extraction tasks.

Chapter 5 presents the definition and formalization of the proposed probabilistic methods that aim to address the Region Segmentation problem. Moreover, in this chapter, we extend the proposed methods to address the Baseline Detection and Region Segmentation problems in an integrated way. Consequently, we strongly suggest reading this chapter after Chapter 4.

Chapter 6 describes the method designed to address the Reading Order Determination problem. This chapter can be read independently of the previous two chapters. However, we encourage the reader to follow a sequential order of these three chapters in order to procure a global vision of the problem and the proposed solutions.

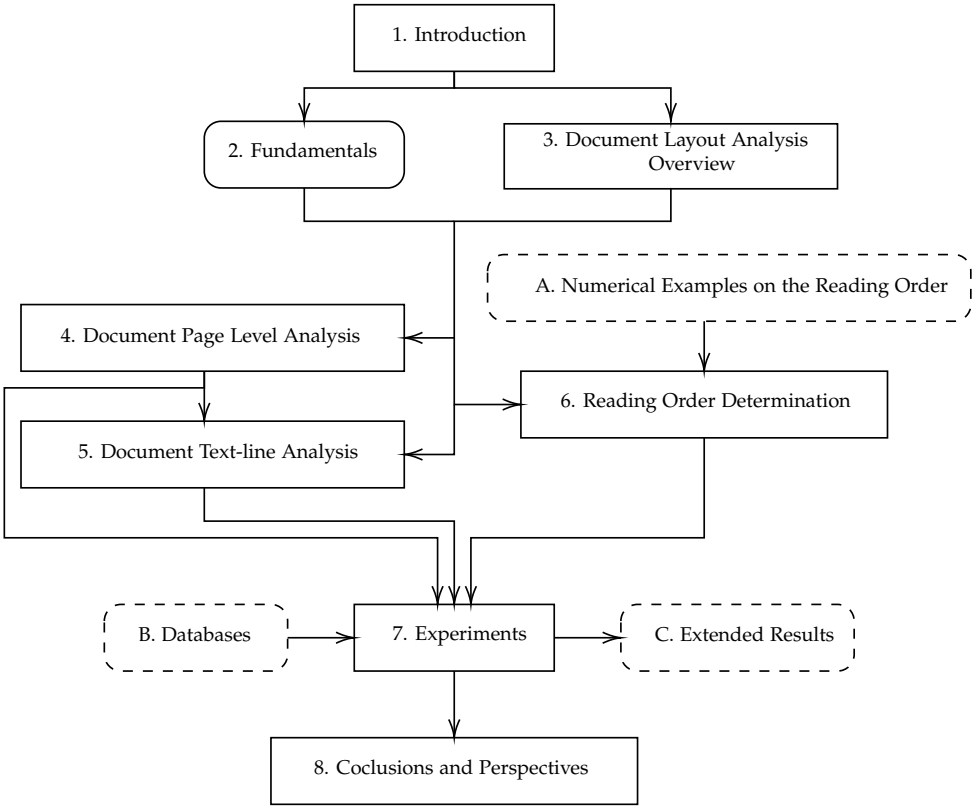


Figure 1: Dependency diagram between the chapters of this thesis.

Afterward, Chapter 7 contains all the experimental evaluations of all proposed methods and algorithms. The experiments are presented in the same order as the task they aim to address were given. For the sake of brevity, we consign the description of the databases to Appendix B, while only a brief summary of them is provided in Chapter 7. Moreover, we restrict the results presented in this chapter to the main metrics and models. Nevertheless, in Appendix C, we offer an extended set of results.

Finally, Chapter 8 summarizes the contributions of this work, including the scientific publications, generated databases and open-sourced software resulting from this work, followed by an outlook of interesting future lines of research in the field.

Abbreviations

Notation	Description
ABP	READ ABP Table.
ANN	Artificial Neural Network.
ATR	Automatic Text Recognition.
Bozen	Bozens Ratsprotokolle.
cBAD-17	Competition on Baseline Detection in Archival Documents 2017.
cBAD-19	Competition on Baseline Detection in Archival Documents 2019.
CC	Connected Component.
CCL	Connected Component Labeling.
CNN	Convolutional Neural Network.
CRF	Conditional Random Field.
CV	Computer Vision.
DLA	Document Layout Analysis.
DT	Decision Theory.
FCR	Filand Renovated District Court Records.
HMM	Hidden Markov Model.
HMR	Handwritten Music Recognition.
HTR	Handwritten Text Recognition.
KWS	Key Word Spotting.
LM	Language Model.

Notation	Description
ML	Machine Learning.
MLP	Multilayer Perceptron.
MTL	Multitask Learning.
OCR	Optical Character Recognition.
OHG	*Oficio de Hipotecas de Girona*.
PLC	Piecewise Linear Curve.
PR	Pattern Recognition.
PT	Probability Theory.
RFC	Random Forest Classifier.
RoI	Region of Interest.
RPG	Region Proposal Generator.
RPN	Region Proposal Network.
SGD	Stochastic Gradient Descent.
SOTA	State-of-the-Art.
VORAU-253	Vorau Abbey library Cod. 253.

Contents

1 *Introduction*

During the long existence of humankind, one of our major needs is to preserve, enhance and share the knowledge acquired by one individual to the others in society. It does not matter the complexity of the knowledge, from essential pieces of knowledge like which plants we can eat and which ones are poisonous, how to make fire, or where to look for shelter on a cold day, to very complex knowledge such as how to build a precise clock or harvest power from nuclear fusion. Once this knowledge is obtained, we (as a society) do not want to lose it; we need to preserve it, share it with other individuals and use it to enhance our lives and generate more knowledge.

For thousands of years, the information needed to preserve that knowledge was shared from one individual to another using face-to-face methods (verbal messages, hand signs, etc.), but this method is very slow and susceptible to errors. So then, information sharing evolves to the use of pictograms, so they can be preserved for generations with less degradation (even today, we can see several pictograms in prehistoric caverns around the world). However, regardless of the improvement to preserve the information obtained with pictograms, it is clear that it is challenging to share it with others.

Later on, the invention of writing and afterward the use of alphabets (2000 BCE) revolutionized it by providing a lasting method to preserve information that can be read and spread easily. Equally important is the use of different materials to write, specifically light materials like papyrus, leather and many kinds of paper, making it possible to share the information from one place to another.

Writing on light materials became the primary vehicle to preserve and share information until the invention of digital systems such as the radio, the TV, digital computers and the Internet. In the current digital era, information can be recorded, stored and shared more cheaply and reliably than ever before. A great example is the Internet, where anyone connected to the network can retrieve or share any piece of information available, even if the physical medium where it is stored is placed on the other side of the world. Despite the improvements in the digital era, it is estimated that the amount of original information recorded in manuscripts and printed documents, since he invention of writing to our days, is greater than the original information recorded in digital systems. For this reason, the need to

digitize the collections of documents available in libraries and archives arises, so the information recorded in those can be accessed, analyzed and preserved, taking profit of the advantages of the digital systems.

In this direction, in the late 1950s, a new research area called Optical Character Recognition (OCR) made significant progress in the transcription of printed documents. An image of a document is processed segment by segment (initially containing only a word and later a complete line of text) to obtain the set of characters present on each fragment and encoded using modern digital formats like ASCII or Unicode. More recently, during the 1970s and 1980s, given the advances in OCR, scientists started to apply the ideas of OCR to handwritten documents. Nevertheless, given the complexity and variability of handwritten documents, the development of new methods became more complex and specific for the task, to the point that they were grouped in a new research area by its own called Handwritten Text Recognition (HTR).

Despite both research areas diverge at the beginning because the documents they aim to address are different, nowadays, given the success of HTR methods on very complex documents[Qui+18a; Sán+19], they are widely used on printed documents as well.

Although both OCR and HTR technologies have evolved to provide good transcripts, these solutions are currently not capable of providing transcripts with the necessary structure to convert them into information. Typically, OCR and HTR need a previous step to extract the structure of the document and segment it in meaningful fragments that can be processed.

This previous step is called Document Layout Analysis (DLA). It does not only provide the document fragments to Automatic Text Recognition (ATR) systems (OCR, HTR, etc) but also the structure of the data present in those documents. Hence, after the transcription process, the transcripts can be rearranged according to the original layout. So the information in the document can be fully extracted (e.g., even if the transcription of several text-lines is perfect, they do not make sense until they are arranged in the correct order; also, the position of the text in the original document can give to the reader a lot of information that is not explicitly written down in words).

Precisely, this has been one of the main motivations driving the development of this thesis. That is, to improve and develop new technology that allows the extraction of the structural information of the documents and, together with ATR technologies, allows libraries and archives to preserve and share the information and knowledge stored in their collections of documents efficiently and robustly using modern digital technologies.

In particular, in this dissertation, we focus on addressing three of the main subproblems of DLA. First, we address the problem of Baseline Detection from a

probabilistic point of view. To that end, two main approaches are explained and evaluated experimentally on textual and musical handwritten documents.

Then, we address the Region Segmentation problem, as explained in Chapter 5, which is a fundamental step to consolidate the information extracted from a document in its correct context. We also formalize two probabilistic approaches to address this problem, and evaluate then on textual and musical documents. Moreover, taking into account the high computational resources employed by the proposed methods, we extend and evaluate the proposed methods in an integrated approach, on which both Baseline Detection and Region Segmentation tasks are addressed together by the same model.

Finally, in order to offer cohesion to the information to be extracted from the processed documents, we address the Reading Order Determination problem. Determining the reading order of the layout elements in a document offers us the ability to contextualize the extracted information (e.g., the transcripts of several text lines) and assemble it in a meaningful way. To that end, we propose and formalize the Reading Order Determination problem as a pairwise probabilistic sorting problem, where instead of predicting the absolute position of some layout element in an ordered set, we aim to obtain the most probable position given the relative relationship between such element and any other layout element in the document. Accordingly, we formalize and evaluate empirically two different decoding algorithms used to obtain the most probable reading order from a set of local relationships between elements.

2 Fundamentals

This chapter provides an overview of several fundamental techniques, methods and algorithms used throughout this dissertation. The most important characteristics of each one are introduced here. However, it is not intended to be exhaustive. Instead, we will refer the interested reader to the publications in the bibliography for further reference.

2.1 Image Processing

As stated in the previous chapter, we aim to obtain information from documents that are digitized into images. Hence, classical image processing techniques will play an important role in most methods developed towards its analysis. Nevertheless, in this dissertation, we will restrict the use of deterministic image processing techniques as pre-processing or post-processing tools, while the analysis itself is consigned to probabilistic techniques.

The following techniques will be proven very useful across this dissertation.

2.1.1 Connected Components Labeling

Connected Component Labeling (CCL) is defined by [Sha96] as the operation over a binary image "that groups the pixels into regions, such that adjacent pixels have the same label, and pixels belonging to distinct regions have different labels". That is, in a binary image, two adjacent pixels are defined to belong to the same regions if they both share the same value (e.g., both are equal to 1), and each different region is known as a Connected Component (CC).

Several efficient algorithms have been developed for CCL, for those we refer the reader to [Gra+16] for a comprehensive benchmark on the topic.

Notice that each CC can represent a layout element in a page image. Hence, it is convenient to use a border following algorithm, like the one presented in [SA85], to extract the contour of the CCs found and represent them by a set of polygons.

2.1.2 Geometric Image Transformations

Geometric transformations modify the spatial relationship between pixels in an image. For instance, an image can be rotated, scaled, translated or sheered with respect to some point or axis.

It consists of two basic operations. First, a spatial transformation of coordinates is performed, for example, an affine transformation (see Section 2.1.2.1) or an elastic transformation (see Section 2.1.2.2). Then, an intensity interpolation operation, like nearest-neighbor interpolation or bilinear and bi-cubic algorithms, is applied to assign intensity values to the spatial transformed pixels [GW08].

2.1.2.1 Affine Transformations

Affine transformation [Wol94] is one of the simplest yet powerful spatial coordinate transformations. It has the general form:

$$\begin{bmatrix} i & j & 1 \end{bmatrix} = \begin{bmatrix} v & w & 1 \end{bmatrix} \mathbf{T} = \begin{bmatrix} v & w & 1 \end{bmatrix} \begin{bmatrix} t_{1,1} & t_{1,2} & 0 \\ t_{2,1} & t_{2,2} & 0 \\ t_{3,1} & t_{3,2} & 1 \end{bmatrix} \tag{2.1}$$

where v, w are pixels coordinates in the original image, and i, j are the corresponding coordinates in the transformed image.

It is important to notice that depending on the values of $\mathbf{T}$, this transformation can perform the rotation, scale, translation and sheer operations simultaneously.

2.1.2.2 Elastic Transformations

Affine transformations perform a constant spatial transformation, where all points in the image are transformed by the same matrix $\mathbf{T}$. Instead, Elastic transformations perform a different spatial transformation for each point.

A common elastic transformation is defined in [SSP03] as a set of soft random displacements as:

$$\begin{bmatrix} i & j \end{bmatrix} = \begin{bmatrix} v & w \end{bmatrix} + \begin{bmatrix} \delta_v & \delta_w \end{bmatrix} \star \mathcal{N}((\delta_v, \delta_w), \sigma) \tag{2.2}$$

where δ_v, δ_w are selected randomly from a uniform distribution in the interval $[-1, 1]$, and $\mathcal{N}(\cdot, \sigma)$ is a Gaussian filter with standard deviation σ.

2.2 Machine Learning

Machine Learning (ML) is a field of computer science and applied statistics that uses and develops computer systems that are able to learn and adapt complicated

functions [GBC16] without following explicit instructions by using algorithms and statistical models to analyze and make predictions from patterns in data.

ML algorithms all aim to learn and improve their accuracy from data (experience). According to [Mit97], "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E".

One common way to classify the ML algorithms is by what kind of experience (data) they are allowed to have during the learning process [GBC16]. In fact, two of the most known categories of ML are: *unsupervised* or *self-supervised* learning algorithms, where the learning algorithm experiences only a dataset containing training samples, and *supervised* learning algorithms that experience a dataset containing training samples but each sample is associated with the *label* or *target* that the algorithm aims to predict.

In this dissertation, we focus on *supervised* methods, where a previously labeled dataset (called *training data*) is used to learn the parameters of a probabilistic model by means of an ML algorithm. Then, the same model is used to predict the layout of some new previously-unseen page document (called *test data* or *evaluation data*).

One of the main issues in ML is the uncertainty of the data to be analyzed, as most of the time the data is ambiguous and incomplete, which makes it virtually impossible to design a deterministic framework general enough to address the problem in all documents and collections. For instance, it is implausible to have an example of all the possible layouts a page could have, even in the case of a single handwritten collection, the number of layout elements is huge (e.g., different number of paragraphs per image, different number of text lines per paragraph, different shapes of the layout elements, etc).

Probability Theory (PT) [Kol56] provides a consistent framework for the quantification and manipulation of uncertainty [Bis06], which makes it a powerful tool for ML problems. PT allows us to make predictions given all the information available to us, while uncertainty is addressed naturally.

As we mentioned in the previous chapter, we are dealing with scanned document images where DLA can be interpreted as a form of ML problem[1] on which the intrinsic structure of the data in the documents is unknown. Hence, under probabilistic formulation, we want to obtain the layout that is most likely to explain the structure of the data recorded in such images.

This is, a supervised ML learning algorithm attempts to train a probabilistic model to estimate the probability distribution $P(\mathbf{y} \mid \mathbf{x})$ according to some optimization criterion (e.g., maximizing the likelihood function, minimizing the least square

[1]Formally, DLA can be also interpreted as a form Pattern Recognition (PR) problem, but, as many authors [Bis06], in this setting we consider ML and PR as synonyms.

error, etc). Then, Decision Theory (DT) [Bis06] is used over the trained model to predict the most probable *target* $\mathbf{y}$ from some input sample $\mathbf{x}$.

2.3 Artificial Neural Networks

During the last decade, Artificial Neural Networks (ANNs) have become the *de facto* standard to model many ML problems. In particular, Convolutional Neural Networks (CNNs) are widely used to model image-related problems.

An ANN is fundamentally composed of interconnected units called artificial *neurons*. An artificial *neuron*, similarly to the real cells present in biological organisms, is interconnected to other neurons and produces an output that is dependent on the given inputs from the other neurons.

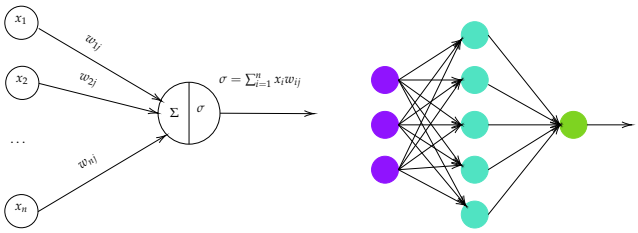
The term "neural network" has its genesis in the work of [MP43], where it is defined as a parametric linear combination of the inputs and a non-linear *activation function* σ. Figure 2.1a shows a diagram of the mathematical model as a single artificial *neuron* with inputs x_i, parameters w_{ij} and activation function σ. Typically, these parameters are called *weights* and neurons also could have a constant input called the *bias*[2].

In 1958 the first algorithm to adjust the weights of the neurons (typically called learning algorithm), was introduced by [Ros58] for binary classifiers. Later on, the *back-propagation* algorithm was introduced by [RHW86] as a general algorithm to help to adjust the parameters to minimize a measure of the difference between the actual output of the network and the desired output (see Section 2.3.1 for details).

A single neuron is useful to solve small problems. But, in order to solve complex problems, the neurons are grouped into *layers* and each layer is connected to other layers. In general, the layers connected to the input and output signal are called *input* and *output* layers, respectively, while the layers connected to neither are called *hidden* layers. Figure 2.1b shows a diagram of a multilayer neural network with one *hidden* layer. This kind of network is called a fully connected multilayer network, since each neuron in a layer is connected to all the neurons in the next layer. Also, this kind of network is widely known as Multilayer Perceptron (MLP), although it is not composed of perceptrons but layers of logistic regression models.

One of the most important design decisions about ANNs is the *activation function* σ. It has been proven that for several activation functions used with a multilayer neural network the universal approximation theorem holds. This means that a multilayer neural network can represent a wide variety of interesting functions when given appropriate weights and activation functions. For instance, it was proven

[2] In this work we omit the bias for clarity on the notation.



(a) (b)

Figure 2.1: Diagram of an artificial neuron (Figure 2.1a) and of a multilayer neural network (Figure 2.1b) with three input neurons, one hidden layer and one output neuron.

by [Cyb89] for *sigmoid* activation functions, which is defined as:

$$\sigma(\mathbf{z}) = \frac{1}{1 + \exp(-\mathbf{z})} \tag{2.3}$$

Furthermore, different activation functions can be used to obtain different results. For instance, the hyperbolic tangent (Equation (2.4)) is often used in regression problems, while other functions such as the Rectifier Linear Unit (called ReLU and denoted by R, Equation (2.5)) [Jar+09] and Leaky ReLU (denoted by L_a, Equation (2.6)) [MHN13], were designed to help the learning algorithm to perform better in practice. These latter functions are normally used only in the input or hidden layers of the ANN.

$$\tanh(\mathbf{z}) = \frac{\exp(\mathbf{z}) - \exp(-\mathbf{z})}{\exp(\mathbf{z}) + \exp(-\mathbf{z})} \tag{2.4}$$

$$R(\mathbf{z}) = \max(0, \mathbf{z}) \tag{2.5}$$

$$L_a(\mathbf{z}) = \begin{cases} \mathbf{z} & \text{if } \mathbf{z} > 0 \\ a \cdot \mathbf{z} & \text{otherwise} \end{cases} \tag{2.6}$$

In the case of classification problems, the output of the ANN is trained to represent the posterior probability of a label (i.e., the class) given the input data. For this kind of problem the *softmax* function [Bri90] is used:

$$y_i = \phi(z_1, \ldots, z_n) = \frac{\exp(z_i)}{\sum_{j=1}^{n} \exp(z_j)} \tag{2.7}$$

where y_i is the i-th output neuron of the ANN.

2.3.1 Training Process

ANNs, like many other ML algorithms, can be trained simply by combining an specification of a dataset, a cost function, an optimization procedure and a model (i.e., its parameters). Moreover, since any of these components are mostly independent of the others, we can use a broad range of training algorithms [GBC16].

One of the most common combinations of these components includes the use of Stochastic Gradient Descent (SGD) [BB08] to minimize a cost function $J(\boldsymbol{\theta})$ parameterized by a model's parameters $\boldsymbol{\theta}$ by updating the parameters in the opposite direction of the gradient of the cost function $\Delta J(\boldsymbol{\theta})$ w.r.t. the parameters. It is usual to combine SGD with the *back-propagation* algorithm [RHW86]. While SGD is used to update the parameters with respect to the gradient, *back-propagation* allows the information from the cost function to flow backward through the network to compute the gradient.

SGD, like many other optimization procedures, relies on the cost function to determine the optimization process. Hence, it is of significance to define a cost function aligned with the goal of the model and that it facilitates the optimizer's work as much as possible (globally continuous and differentiable). Since most of the time it is not possible to define a cost function with all the desirable mathematical properties, the affinity of the selected function should be evaluated empirically.

For instance, in the case of a binary classification problem[3], let $\Phi(\mathbf{x}, \boldsymbol{\theta}), \Phi : \mathbb{R}^d \rightarrow (0, 1)$ be the output of an ANN, whose activation function on the output layer is a logistic sigmoid. Then, $\Phi(\mathbf{x}, \boldsymbol{\theta})$ can be interpreted as the conditional probability of the target $y \in \{0, 1\}$ given the input $\mathbf{x} \in \mathbb{R}^d$, restricted by the model parameters $\boldsymbol{\theta}$. Moreover, we want to estimate the set of parameters $\boldsymbol{\theta}$ that minimize the expected dissimilarity between the empirical distribution, defined by the training data, and the model distribution measured by the Kullback-Leibler (KL) divergence. Nonetheless, it can be demonstrated [Bis06] that minimizing this KL divergence corresponds exactly to minimize a *cross entropy* cost function between those distributions, defined as:

$$J(\boldsymbol{\theta}) = -\sum_{n=1}^{N} y_n \log(\Phi(\mathbf{x}_n, \boldsymbol{\theta})) + (1 - y_n) \log(1 - \Phi(\mathbf{x}_n, \boldsymbol{\theta})) \tag{2.8}$$

where N is number of samples of the training set $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ with target values $Y = \{y_1, y_2, \ldots, y_N\}$.

Furthermore, in the case where we have M independent[4] binary classifications (i.e., $\mathbf{y}_n = \{y_{n,1}, y_{n,2}, \ldots, y_{N,M}\}$) to perform over the same input $\mathbf{x}_n$, then we can

[3] In this chapter, we focus on *discriminative* models. Nonetheless, we refer the reader to [Bis06; DHS01; GBC16] for a comprehensive analysis on *discriminative* and *generative* models.

[4] Or at least assumed independent

have an ANN having M outputs (all with logistic sigmoid activation functions), and the corresponding *cross entropy* cost function becomes:

$$J(\boldsymbol{\theta}) = -\sum_{n=1}^{N}\sum_{m=1}^{M} y_{n,m}\log(\Phi_m(\mathbf{x}_n,\boldsymbol{\theta})) + (1-y_{n,m})\log(1-\Phi_m(\mathbf{x}_n,\boldsymbol{\theta})) \tag{2.9}$$

where $\Phi_m(\cdot)$ is the output of the network for the classifier m.

Finally, if we consider the classical multinomial classification problem, where each input is assigned to one of K mutually exclusive classes, and *softmax* activation function is used in the output layer of the ANN. Then, a similar analysis to the binary setting can be carried out to obtain the following *cross entropy* cost function:

$$J(\boldsymbol{\theta}) = -\sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{k=1}^{K} y_{n,m,k}\log(\Phi_{m,k}(\mathbf{x}_n,\boldsymbol{\theta})) \tag{2.10}$$

where $\mathbf{y}_{n,m} \in \{0,1\}^K$ is a one-hot representation of the target variable, and $\boldsymbol{\Phi}_m : \mathbb{R}^d \to [0,1]^K$.

Nonetheless, as the optimization process does not change when we rescale the cost function, it is very common to normalize the cost function in terms of the number of samples (N) and predictions performed (M), which lead us to a version of the optimization criterion expressed as an expectation with respect to the empirical distribution defined by the training data [GBC16]. For instance, Equation (2.10) becomes:

$$J(\boldsymbol{\theta}) = \frac{-1}{N}\sum_{n=1}^{N}\frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} y_{n,m,k}\log(\Phi_{m,k}(\mathbf{x}_n,\boldsymbol{\theta})) \tag{2.11}$$

Furthermore, we refer the reader to [Bis06] for a detailed analysis on training neural networks on regression problems as well.

As mentioned before, ANNs have performed remarkably well on many ML tasks. However, they tend to over-fitting[5] the training data which can be avoided by using big amounts of data. Unfortunately, many applications do not have access to the required amount of labeled data. As a result, several regularization techniques have been proposed [KGC17] to discourage learning a very complex model and avoid over-fitting. For instance, it is usual to combine the cost function with a regularization term [GBC16], also to circumvent the limited access to training data, data augmentation [SK19] techniques such as affine transforms (see Section 2.1.2.1) and elastic transforms (see Section 2.1.2.2) are very common.

[5]Over-fitting happens when the models had fitted so much to the training data that it is unable to provide accurate predictions for new data from the same distribution.

2.3.2 Convolutional Neural Networks

Although the capacity of MLPs is well-established and proved, in practice, its potential is restricted for high-dimensional data. Since all neurons in a layer are connected to all neurons in the next layer, the number of parameters is very high, which becomes a problem for training purposes as we need a model with a finite number of parameters. Furthermore, since each input unit would be connected to all pixels in the input image, we would need n^2 parameters for an input of size n. This makes the fully connected layers slow when processing high dimensional data, and highly *overparameterized*, which may create or aggravate the problem of over-fitting.

A basic solution to this dilemma is to manually define a set of features using a non-linear transformation of the input data to reduce its dimensionality, hoping those features to be a good representation of the data, so with fewer layers and units, an ANN could make good predictions. Nevertheless, it is challenging to design those features manually.

Another approach, used in data with grid-like topology (like images), is to replace some hidden layers in an ANN with convolutional layers and let the network to automatically learn those features, those ANNs are called Convolutional Neural Networks (CNNs) [LeC89].

Convolutional layers are similar to fully connected layers, but each neuron is only connected to the neighbor units from the previous layer instead of to all of them. Additionally, all units share the same parameters, which is a crucial property, as the number of parameters does not depend on the size of the input data but only on the size of the neighborhood around each unit (i.e., the receptive field).

Formally, a convolutional layer is simply a layer that makes use of the convolution operator (typically denoted as $*$) to obtain a shift-invariant [LeC89] representation of the input data. Let $\mathbf{x}$ to be a two-dimensional input signal, a convolutional layer will learn a set of parameters $\mathbf{w} \in \boldsymbol{\theta}$ (called a kernel) and generates an output signal $\mathbf{y}$ (called feature map), as depicted in Figure 2.2, by convolving $\mathbf{x}$ and $\mathbf{w}$:

$$\mathbf{y} = \mathbf{x} * \mathbf{w} = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{x}) \odot \mathcal{F}(\mathbf{w})) \tag{2.12}$$

where $\mathcal{F}(\cdot)$ is the Fourier transform and $\odot$ the point-wise product. Furthermore, in the case of two-dimensional images[6], along with a two-dimensional kernel:

$$y_{i,j} = (\mathbf{x} * \mathbf{w})_{i,j} = \sum_m \sum_n x_{i,j} w_{(i-m),(j-n)} = \sum_m \sum_n x_{(i-m),(i-n)} w_{i,j} \tag{2.13}$$

where m and n range in the size of $\mathbf{w}$, and the last equality holds due to the commutative property of the convolution operator.

[6] In practice, images are typically a multi-channel tensor (e.g., RGB), in that case the same formulation holds, but another sum must be added to Equation (2.13) to take into account that new dimension.

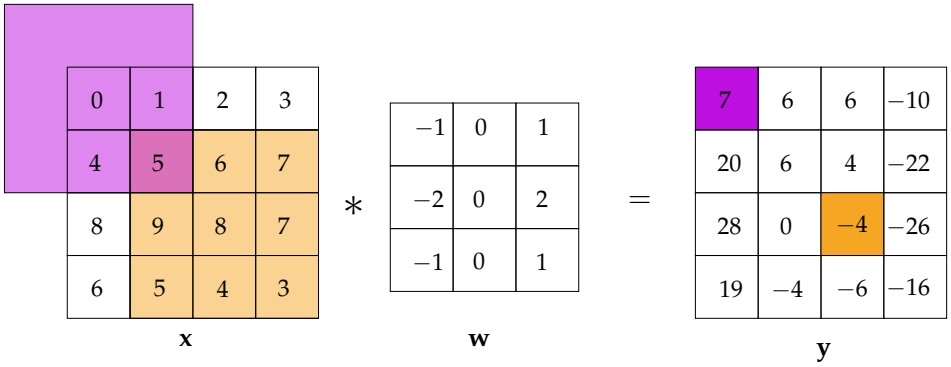


Figure 2.2: Diagram of the two-dimensional convolution operator. The two-dimensional input signal $\mathbf{x}$ is convolved with the weights matrix $\mathbf{w}$ to obtain the output signal $\mathbf{y}$.

Although the commutative property of the convolution operator arises naturally from the definition, during implementation it involves flipping the kernel relative to the image, which is computationally expensive but irrelevant to the CNN. Consequently, many neural networks libraries implement a related function called *cross-correlation* ($\star$), which is similar to the convolution but without flipping the kernel:

$$y_{i,j} = (\mathbf{x} \star \mathbf{w})_{i,j} = \sum_m \sum_n x_{(i+m),(i+n)} w_{i,j} \tag{2.14}$$

Notice that for pixels in the border of $\mathbf{x}$, the convolution operator would need some "pixels" laying outside the input data (e.g., the red regions in Figure 2.2). For those cases, two main approaches are very common in practice, first, those pixels in the border are just ignored, this approach will lead to an output signal $\mathbf{y}$ smaller than the input signal, which in many applications is desired. The second approach is to use a *padding* value for those "pixels" laying outside (usually 0) which has the advantage of using all the values of the input signal and could keep the size of the output signal equal to that of the input.

Very often it is desired to use convolutional layers to obtain an encoded representation of the input signal with reduced size (dimensionality). To that end, several strategies can be used, for instance using *padding* smaller than half of the kernel size, or more commonly using a pooling layer [LeC+90]. Also, the convolution operator can be seen as a sliding dot product, with a stride of one data point at the time, between the input signal and the kernel, hence, increasing the stride (commonly to 2) will produce smaller volumes spatially.

2.3.2.1 Transposed Convolutional Layer

As the convolution operator has proven to be useful to obtain a good representation of the input signal, the next logical step is to obtain an inverse of that representation, so we can obtain the original signal (or at least a similar one). But, obtaining it is not trivial, for instance, in the example of Figure 2.2 there are infinity number of input signals $\mathbf{x}$ that when convolved with that specific kernel $\mathbf{w}$ they generate the output signal $\mathbf{y}$.

For this reason, rather than using the inverse of the convolution operator, it is useful to use another operator that goes in the opposite direction of the convolution operator,i.e., from $\mathbf{y}$ to something with the same size and shape of $\mathbf{x}$, while maintaining a connectivity pattern that is compatible with said convolution. This operation is called transposed convolution[7], and it does not guarantee to recover the input itself, but gives us the formal framework to obtain a signal with the same shape as the input.

To understand the transposed convolution ($\overline{\star}$) it is useful to go back to the convolution operator and rethinking it in its matrix form. If we unroll the kernel $\mathbf{w}$ from left to right, top to bottom, it could be represented as a sparse matrix $\mathbf{c}$. For instance, in the case of Figure 2.2:

$$\begin{aligned}\mathbf{c} &= \begin{bmatrix} w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 & 0 \\ 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} & 0 \\ 0 & 0 & 0 & 0 & 0 & w_{0,0} & w_{0,1} & w_{0,2} & 0 & w_{1,0} & w_{1,1} & w_{1,2} & 0 & w_{2,0} & w_{2,1} & w_{2,2} \end{bmatrix} \\ &= \begin{bmatrix} -1 & 0 & 1 & 0 & -2 & 0 & 2 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & -2 & 0 & 2 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & -2 & 0 & 2 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & -2 & 0 & 2 & 0 & -1 & 0 & 1 \end{bmatrix}\end{aligned} \tag{2.15}$$

Similarly, by flattening the input signal $\mathbf{x}$ we obtain:

$$\begin{aligned}\mathbf{x}' &= \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} & x_{1,0} & x_{1,1} & x_{1,2} & x_{1,3} & x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} & x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{bmatrix}^T \\ &= \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 8 & 7 & 6 & 5 & 4 & 3 \end{bmatrix}^T\end{aligned} \tag{2.16}$$

Then the convolution can be seen as the multiplication between those matrices:

$$\mathbf{y}' = \mathbf{c}\mathbf{x}' \tag{2.17}$$

where $\mathbf{y}$ can be obtained by de-flatten $\mathbf{y}'$.

Now, it is straightforward to see that if we multiply both sides of Equation (2.17) by the transposed version of $\mathbf{c}$ we can recover the *shape* of the input signal ($\mathbf{x}'' = \mathbf{c}^T\mathbf{y}'$).

Note that using this transposed convolution with $\mathbf{c}^T$ does not guarantee to recover the input itself, as it is not defined as the inverse of the convolution, but rather just returns a feature map that has the same size and shape [DV18]. For this

[7] Also called *fractionally strided convolution* or *deconvolution*. Although the therm deconvolution is formally incorrect and it should be avoided.

reason, instead of using $\mathbf{c}^T$ directly, it is very common to allow the CNN to learn the values of the transposed kernel in the same way as it learns the values of the original kernel. Consequently, we can obtain a new kernel that best fits the requirements of the task we aim to solve.

In Figure 2.3, we depict the process of implementing the transposed convolution, where we use $\mathbf{y}$ as the input of the transposed convolution and $\mathbf{x}$ as its output for consistency with the convolution operator, although normally in the literature it is common to see the transposed convolution independently of the convolution and use $\mathbf{x}$ as input and $\mathbf{y}$ as output.

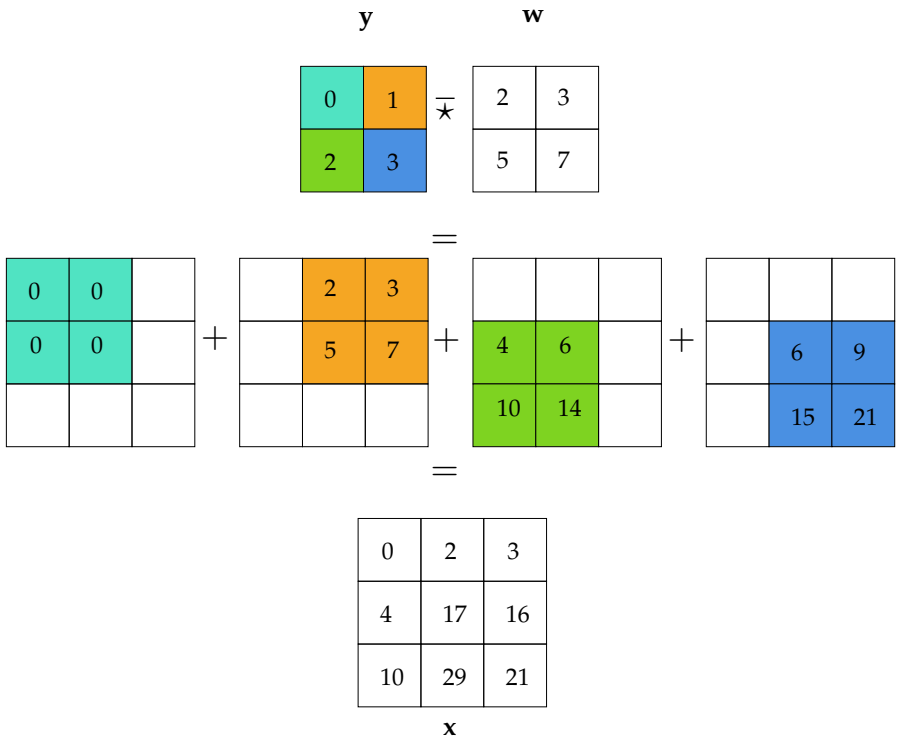


Figure 2.3: Diagram of the two-dimensional transposed convolution operator. The input signal $\mathbf{y}$ is processed along with the weights matrix $\mathbf{w}$ to obtain the signal $\mathbf{x}$.

For a comprehensive explanation of convolution and transposed convolution operators and how they are used in Deep Learning, we refer the reader to Dumoulin and Visin's guide to convolution arithmetic [DV18].

2.3.3 Region Proposal Network

In many applications it is common to have more than one region of the input signal where we are interested in making some predictions. For instance, in image segmentation, we aim to obtain as many predictions as objects are present in the input image.

One approach to obtain several predictions is to split the input image into a set of smaller regions and use an ANN to obtain a prediction of each one. Although this may be useful in some specific applications, in general it is difficult to split the input image in a useful way.

Searching for a "'useful way" to split an image is a huge problem. For instance, a Brute Force algorithm should check for $\sum_{k=1}^{n} \binom{n}{k}$ possibilities, where n is the number of pixels in the input image. Consequently, several methods have been developed to alleviate this issue. First, *sliding window* algorithms, like [VJ01], have been proposed to cope with the complexity of the problem, but still requires to process a lot of regions per image (in the order of 10^6 for multi-scale predictions). Then, object proposal algorithms have emerged, often reducing the number of regions by merging them [San+11] or by filtering out undesirable regions based on some score [ADF10]. However, it is very difficult to define a general merging or scoring procedure. Therefore, new methods aim to train an algorithm to automatically propose a set of regions where it is more probable to make a meaningful prediction [Erh+14; Ren+16].

There is a large literature on object proposal methods [Hos+16], however, here we focus on the Region Proposal Network (RPN) architecture [Ren+16] due its demonstrated versatility and relatively low computational requirements.

A RPN is a CNN that scans the feature maps generated by a previous CNN (called the *backbone*) by a sliding window, simultaneously regressing region bounds and scoring the membership of each region to a set of object classes[8]. On each location of the sliding window, k region proposals are predicted, parameterized relative to a set of k reference boxes or anchors. Typically, anchors span a range of scales and aspect ratios (e.g., 32, 64, 128 and $1:1$, $1:2$, $2:1$).

Then each region proposal is filtered by its scores and a non-maximal suppression algorithm is used to handle overlaps. Finally, the region proposals that remain are called Regions of Interest (RoIs) and passed to the next stage.

[8]Membership is usually understood as a binary variable that is 1 if the region proposed belongs to an object in the image or 0 if it belongs to the background.

3 *Document Layout Analysis Overview*

Informally, DLA is the problem of searching for the intrinsic structure of the documents. It aims to extract every structural element of the document and the relationships that exist between them. For instance, we want to know the distribution of the information across the pages of a document: where the lines of text are, how they are arranged into groups (e.g., paragraphs), if there is an illustration and how that illustration is related to the surrounding text, etc. Even informal, this definition help us to understand the complexity of the problem.

Specifically, on handwritten documents, the structure varies from document to document and follows no strict rule but the art of the writer and the circumstances of the time. Also, the physical document is exposed to different degradation processes for many (sometimes hundreds) years. As an illustration, think of a customs officer in the crowded port of Saint Mary in Cadiz, Spain, circa 1770. One of their main duties was to take note of all the goods that were transported by hundreds of ships from America to Spain. They recorded all that information under a lot of pressure, buried in a maze of merchandise, ships and bustling sailors. So, fast writing was the norm, acronyms were frequently used and a clean straight line of text was not imperative. In the Figure 3.1 a page of that time is presented, probably it was written very fast and then exposed to the elements for hundreds of years.

Furthermore, once the document is in our hands, we as humans tend to be inconsistent at defining the layout of a document. For example, when two users are asked to draw the contour of some text line in a document, probably they will end with two different contours, as, for instance, the definition of the border between one text line and the next one is fuzzy. Consequently, for the same document we can obtain several layouts with small differences, and all of them are considered as correct by the users.

For these reasons, a deterministic approach will not be enough to handle the DLA problem in a general and efficient manner. As a matter of fact, a system that aims to address the DLA problem on handwritten documents must be general enough to handle several types of documents (one column, two columns, with a variable number of paragraphs per page, etc.) without having to re-define the proposed solution each time. Also, it should be able to handle the uncertainty of the document structure and the fuzziness of the definition of a "correct" layout.

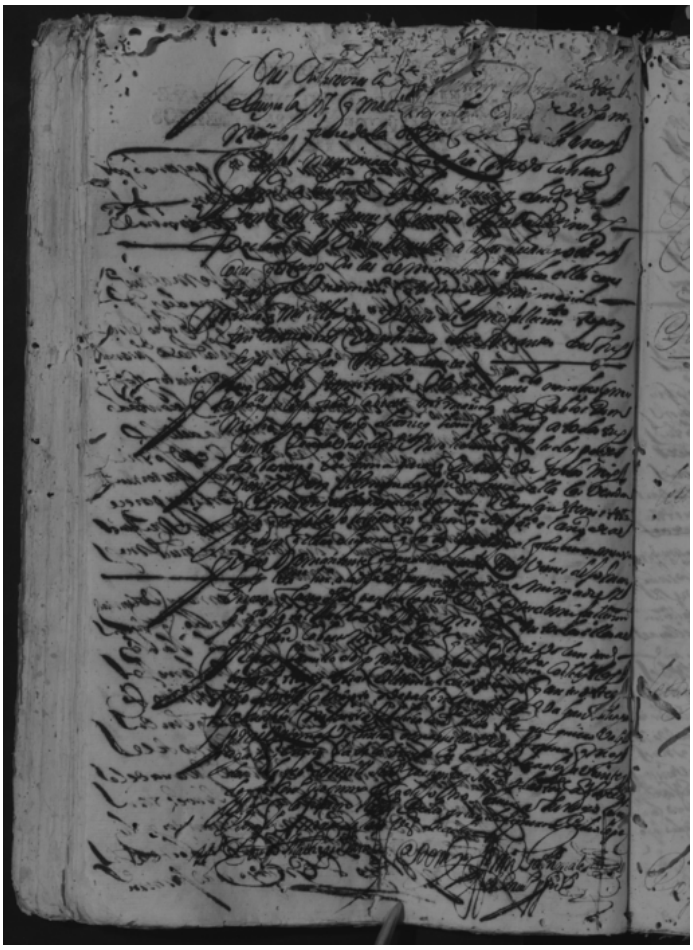


Figure 3.1: Example of a page written circa 1770 in Cadiz, Spain. Severe degradation can be observed, including moth holes and ink bleed-through.

Fortunately, statistical methods provide a framework for expressing such uncertainty in a precise and quantitative manner [Bis06].

As the title of this thesis suggests, we address DLA as a statistical pattern recognition problem. Given a set of digitized document pages, we want to *predict* —under a probabilistic formulation—which structure is most likely to explain the layout of such documents. To that end we apply ML techniques to, in a supervised manner, train a model to learn the probabilistic distribution of the layout given the

documents. Finally, Decision Theory (DT) is used to make optimal[1] predictions based on that probabilistic representation.

3.1 A Note to Clarify Some Concepts

During the execution process of this work, we found some concepts that sometimes are fuzzy or even contradictory in the bibliography of this research area. Here we list some of them and provide the definition we will use throughout this work. We hope this helps to avoid ambiguities or confusion to the reader.

Layout element: we call *layout element* to any object that is part of the layout of some document. For instance, a line of text, a paragraph, an illustration, etc.

Text Baseline: is the imaginary line upon which the line of text rest. It is normally called just "baseline".

Detection: is the process to identify if a layout element is present or not in a document. However, for historical reasons, we will use "baseline detection" to define the process to obtain the Piecewise Linear Curve (PLC) that best fits the text baseline, although it should be called "baseline segmentation".

Segmentation: is a process similar to detection, but we go further and obtain a 2D representation of the layout element. For instance, "region segmentation" is the process of obtaining a polygon (or a mask) that defines where an object is and its shape.

Extraction: is the process from which a layout element is separated from the document. For instance "Text -Line Extraction" refers to the process to extract all the text lines defined in the layout of a document image and generate a set of sub-images, each one containing only a line of text.

3.2 Problem Definition

Document Layout Analysis is well-defined by Cattoni [Cat+98] in 1998 as:

> ... [DLA] is a mix of the so-called *geometric* and *logical layout analysis*. The geometric layout analysis aims at producing a description of the geometric structure of the document. This phase involves several processes:

[1] Under some assumptions and according to appropriate criteria.

> some preprocessing steps and the page decomposition step. This last step aims at decomposing the document image into maximal homogeneous regions whose elements belong to different data types (text, graphics, pictures, ...) [...]. The logical layout analysis aims at identifying the different logical roles of the detected regions (titles, paragraphs, captions, headings) and relationships among them.

Although the problem is defined in two steps, it is clear that they depend on each other. Furthermore, we can express the DLA as the solution to an optimization problem, where we want the most probable layout structure h that explains the intrinsic structure of some collection of documents $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Under Maximum a posteriori probability framework, this is:

$$\hat{h} = \arg\max_{h \in H} \mathrm{P}(h \mid \mathcal{X}) \tag{3.1}$$

where H is the set of all possible layout structures. Also, each layout h encompasses both *geometrical* and *logical* information, and h can be represented in several ways (a graph, a tree, a list of elements, ...) as will be explained in detail in Section 3.3.

Notice that this is a general definition that can be applied to any kind of collection, but this same generality carries the complexity of the collection with it, making the problem very difficult or unsolvable in most cases. Given that, simplifying the problem by using informed assumptions is a *de facto* rule in the field.

In general, in this thesis we will follow a discriminative approach to estimate the posterior probabilities involved in Equation (3.1). Indeed, discriminative models such as ANNs have become the *de facto* standard for classification problems in the last decade.

Nonetheless, it is important to notice that explicit prior information about the layout h, $\mathrm{P}(h)$, could be useful to address the problem. For instance, modeling the problem in a generative way instead of discriminative will allow us to make explicit use of the prior distribution (e.g., modeled by frameworks similar to N-grams [Bos20] or Probabilistic Context-free Grammars [Alv+13]). However, a generative approach is considered to be more complex than the discriminative [Bis06], and the generative approach tends to have a higher asymptotic error (as the number of training samples become large) [NJ02]. Consequently, although here we focus in the discriminative setting, a hybrid approach such as the ANN/Hidden Markov Model (HMM) [Blu15] commonly used in ATR should be considered in a future work.

3.3 Taxonomy of Document Layout Analysis

Although DLA problem is well-defined, its complexity forces scientists to make some assumptions about the problem to simplify it. Dividing it into several specific

sub-tasks that can be more easily addressed than the general problem. Normally, the evolution of these sub-tasks is strongly related to the development of new ATR technologies.

Even though those assumptions are key in order to handle the problem under circumscribed resources, we cannot forget that they become with a price. As Hunt wisely points out in [Hun75]:

> ... Statistical methods are powerful ones which depend on strong assumptions. When these assumptions can be made, they should be. On the other hand, it is important that the nature of the assumptions to be understood, for when they are unwanted their use may lead to very misleading results.

In order to organize and understand DLA methods, we follow a *Problem Taxonomy* that seeks to classify the different methods based on the sub-problem of DLA that they aim to solve (and the assumptions made). As a guide, in Figure 3.2 we provide the hierarchical structure of the taxonomy. For instance, *Text-line Analysis* focuses mainly on text-line segmentation and text-line extraction, hence its output is defined at the text-line level.

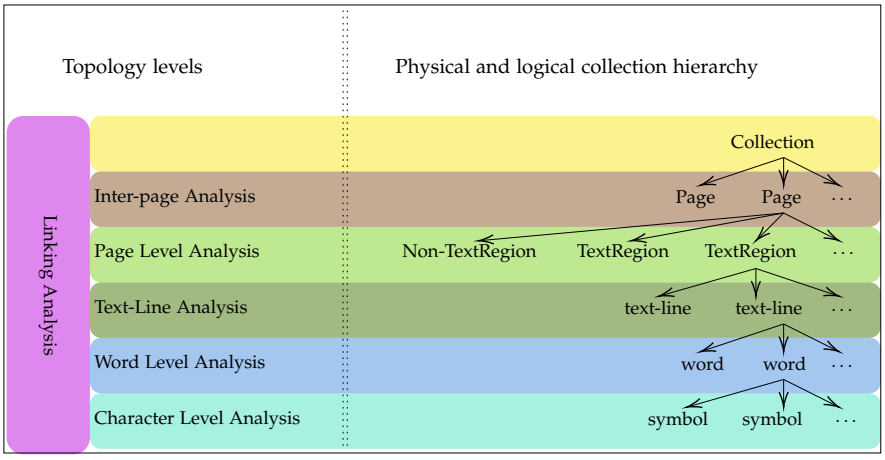


Figure 3.2: Illustration of the structure of the *Problem Taxonomy*. Each taxonomy level represents a different class. Each of the levels focuses on a physical or logical collection level, except Linking Analysis that could focus on one or more levels.

In the following sections, we analyze each of the taxonomy levels, along with the State-of-the-Art (SOTA) methods developed on each one.

3.3.1 Character Level Analysis

This task (also called Character Segmentation) aims to split a previously extracted (normally manually or using a very strict template) fragment of text into a set of smaller fragments, each of them containing a single character (see Figure 3.3). Such systems were developed mainly to provide input to early OCR systems based on symbol-level classifiers. We refer the reader to [CL96] for a detailed survey on the topic.

In this case, h is just a list of coordinates where the input text fragment has to be spitted, as shown in Figure 3.3. Also, each image is assumed to be independent of any other image, and any character independent of the others around it.

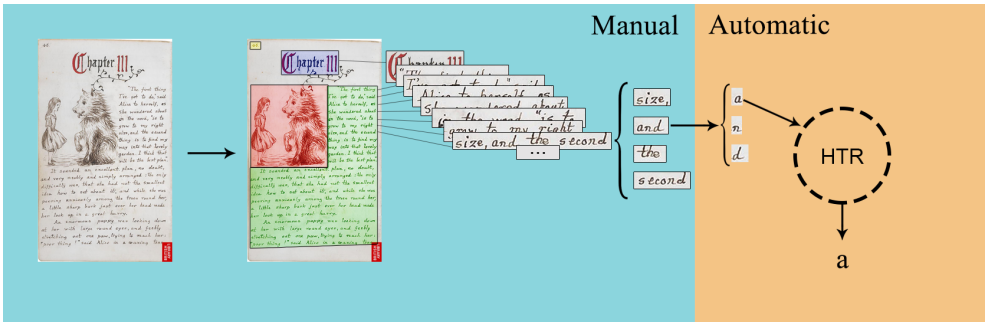


Figure 3.3: Character level segmentation process. Most of the process is done manually, where only words are automatically divided into characters to be recognized.

Although important at the time, this problem is considered solved [Nag00] due to the evolution of ATR methods towards recognition at higher levels where linguistic and syntactic context is available.

3.3.2 Word Level Analysis

Similar to the previous case, in this case, each of the smaller fragments should contain a single word, instead of a character (for this reason it is also called Word Segmentation). Again, a human needs to previously split the input image into a set of lines of text. In this case, the assumptions are relaxed, and even though each image is still assumed to be independent, now the relationship between characters is taken into account.

Most of the process is still handled manually, as shown in Figure 3.4. But, there is no need anymore to segment each word into the characters that compose it.

Most works in ATR published before 1995 assumes that words are previously isolated from the lines of text, or they argue that the process to obtain those words

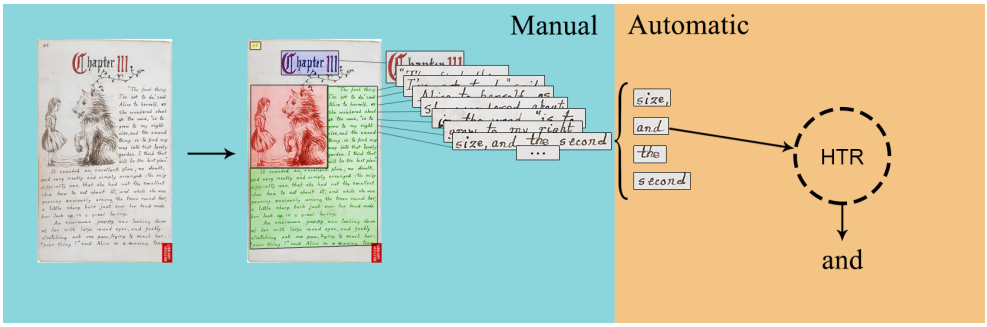


Figure 3.4: Word level segmentation process. Most of the process is done manually, except lines that are automatically divided into words to be recognized.

is trivial [KG98] since words are separated by clear spaces. Which was proven to be false for handwritten text documents, where, in some cases, that "space" does not physically exist or is so small that even humans found it tough to split the words correctly.

Nevertheless, the most common automatic approach to word separation, from those years until now, is based on the following steps: (i) determine the CCs in a given line of text, (ii) compute the distance between pairs of adjacent components [SC94; MN95; KG98], (iii) classify those distances into inter-word gaps and inter-character gaps [KG98; VB05; Lou+09]. Finally, inter-word gaps are assumed to be word separation points.

Although word segmentation is still an active research area, its efforts are addressed to specific applications like some specific Key Word Spotting (KWS) systems [Gio+17], while general ATR applications are moved forward to higher levels like text line or even full page recognition.

3.3.3 Text-line Analysis

This is the most common DLA task performed nowadays, mainly because it allows the ATR systems to process a complete line of text, taking into account the context of each word.

In the literature, it is very common to name the task of searching for the text lines in a document as text-line segmentation, while the process of physically divide the input image into a new digital file for each text line is called text-line extraction.

At this level, the input image can be a full page of text or a pre-segmented paragraph, see Figure 3.5. Also, the relationship between words is taken into account, but each line of text is assumed independent of the others.

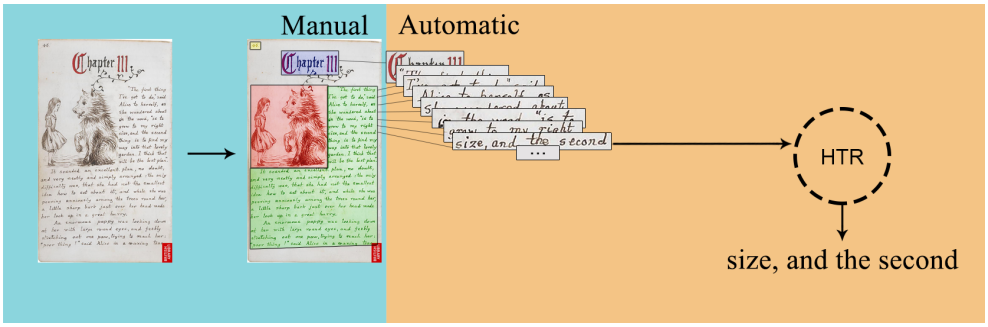


Figure 3.5: Text-line level segmentation process. The main goal is to split each paragraph of text into a set of text-lines, where each paragraph is assumed to be previously segmented.

Nowadays it is very common to handle the text-line segmentation problem as a Baseline Detection problem [Die+17; Die+19]. Baseline Detection aims to detect not the polygon that surrounds a line of text but the imaginary line upon which the line of text rest. This baseline is normally represented by a Piecewise Linear Curve (PLC) composed of few vertices, which makes it a very useful representation to handle manual labeling.

Several methods have been developed to handle the text-line segmentation problem. We can divide those methods into deterministic and probabilistic approaches. First, deterministic methods address the problem as a Computer Vision (CV) problem, using classical CV techniques to (under several assumptions and constraints) segment the input images. Because of the deterministic nature of those approaches, they are designed to work well on documents that comply with the hard-coded underlying set of assumptions and rules that dictate the type of layouts they can be applied to [Bos20], but they are not general enough to handle the diversity of documents we can found on libraries and archives. A very extensive survey on this kind of method is presented in [LZT07], and most recently by [EGO17].

On the other hand, most modern approaches tend to address the problem from a probabilistic point of view. In most of the cases—even if they do not explicitly define it in that way—the goal is to obtain the most probable layout given the input image. For instance, in [BTV12] they train a HMM along with an *n-gram* Language Model (LM) to estimate the most probable sequence of vertical tokens (line, interline, blank-line, non-text) in an input image. Furthermore, this approach is followed by Moysset et al. in [Moy+15], where they train an ANN to estimate the most probable sequence of "line" and "interline" tokens in a paragraph, where paragraphs are assumed to be previously segmented. Despite its applicability to single-column documents or pre-segmented paragraphs, they cannot handle complex layouts.

Furthermore, in recent years, with the widespread use of CNNs, several methods were proposed by us in [Qui+18a; QTV19] and similarly by others [Ren+18; Fin+18; ASK18; Grü+19], to address the baseline detection problem as a two-stage method. In the first stage, a CNN is used to obtain a probability map of each pixel in the input image to be part of a baseline, while in the second stage this probability map is used, in conjunction with a set of heuristics, to detect the set of PLCs that best represents the baselines present in the document.

Those systems assume that each layout element (text-line, baseline) is independent of each other in an image, and that each image is independent of any other in the collection. Notice that, although none of them formalize their approach under PR and DT frameworks, those probabilistic methods aim to solve a sub-problem defined from Equation (3.1) as:

$$\hat{h} = \arg\max_{h \in H} \prod_{\mathbf{x} \in \mathcal{X}} \prod_{e \in h} \mathrm{P}(e|\mathbf{x}) \tag{3.2}$$

where e is an element in the layout h (in this case lines of text or baselines) of each page image $\mathbf{x}$. We will formalize this probabilistic approach in Chapter 4 and analyze its performance in Chapter 7.

Notice that even though those DLA systems are developed to extract all the lines of text on a page, most of the time the reading order or any other kind of relationship between them is left out of the scope of the system. We will also introduce the reading order problem for text lines in Section 3.3.5.1 and formalize a solution to it in Chapter 6.

3.3.4 Page-level Analysis

This task aims to split the page into a set of elements of interest (called region segmentation), taking into account not only the text related elements (paragraphs, page-number, marginalia, etc) but any other kind of region like images, drawings, stamps, or any other kind of data present in the page, see Figure 3.6.

In contrast to the previous task, in this case not only *geometric layout* is performed (segmentation of the page into a set of regions of interest) but to some extent *logical layout* as well (in the form of region labeling). However, the relationship between elements is not taken into account. With this in mind, h becomes not only a set of polygons but a set of polygons with a class attribute $h = \{e_1, e_2, \ldots, e_K\}$ and $e_k = \{\mathbf{p}, c\}$, where $\mathbf{p} \in \mathbb{N}^d$ is the shape definition (i.e., the polygon) and c its class.

Notice that elements defined here (e_k) could be also lines of text as in the previous case so that regions and lines of text can be segmented together by the same formulation. In addition, this sub-problem is commonly addressed using the same formulation as in Equation (3.2), taking into account that e_k is more complex in this case.

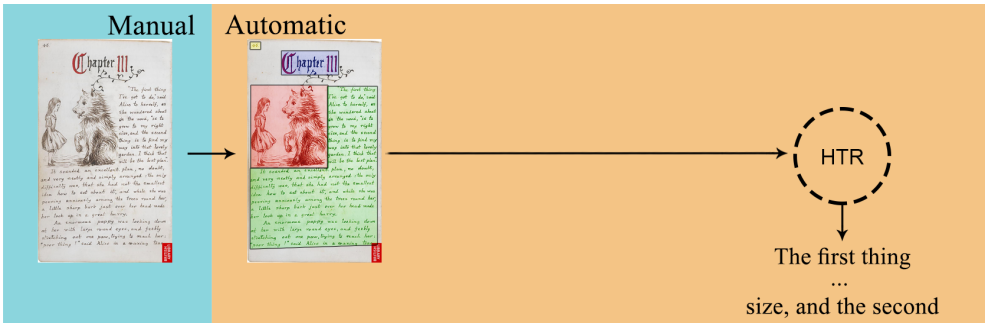


Figure 3.6: Page level segmentation process. The main goal is to split each page into a set of elements of interest. However, the relationship between elements is ignored.

Traditionally, region segmentation and region labeling have been addressed separately. For instance, classical CV methods have been used to address the segmentation problem [CW09], as well as methods based on some kind of pixel-level classifier (MLP [BI11; Wei+13], Gaussian Mixture Models and Support Vector Machines [Wei+13]) whose input is a set of handcrafted features from the input images (Gabor filters, Kalman filters, Connected Components, Multi-scale images, etc). Others aim to provide an interactive framework to review the results of the segmentation algorithm [Qui+17]. On the other hand, some methods aim to provide, to some extent, the correct label to each region previously segmented. Some of them focus only on separating text from non-text regions [ZC15; Wei+13], while others were designed to handle several region types [Coh+13].

Nowadays, it is common to handle region segmentation and region labeling as a single problem, for instance in [FT12] they use Conditional Random Fields (CRFs) along with relative location features to segment the document into three different regions, in [LCC08] they use a multi-resolution strategy to segment up to six different regions. Furthermore, with the extended use of ANNs, more methods aim to address the region segmentation and labeling as a classification problem [ASK18; Qui+18a; QTV19] and using RPNs [Pru+19; SMJ19; Stu+19; ZTY19].

In Chapter 5 we formalize a probabilistic approach to the region segmentation and labeling problem. As they are nowadays addressed together, in this dissertation we will refer to region segmentation as the union of both problems and make no further differentiation between them. Furthermore, we propose to handle this problem, along with baseline detection, in an integrated manner. That is, using a single model to handle both problems at the same time. We provide experimental evidence of its performance in Chapter 7.

3.3.5 Linking Analysis

Linking analysis can be understood as the process to find all the possible relations between layout elements, for instance, name entity, link prediction (caption, figure, reference in text), reading order, table analysis, hyphenation analysis, etc...

Here $e \in h$ is interpreted as a multi-valued set $e = \{\mathbf{p}, c, l\}$, where $\mathbf{p} \in \mathbb{N}^d$ is the shape definition of e, c its class, and l the set of relationships between the element e and any other element $e' \in h$.

In this work, we analyze the reading order, which is one of the most important relationships between elements in a document, since it defines the meaning of the data recorded in a document. For other linking analysis tasks, we refer the reader to [Bor+20; MH16] for name entity, to [JET19] for link prediction, to [Gao+19] for table analysis, and to [VT21] for hyphenation analysis.

3.3.5.1 Reading Order Determination

As commented before, many sub-tasks assume that each element of the document is independent of any other. This is a very hard assumption that prevents ATR systems to fully use the context of the elements to improve document recognition and to present the data in the correct order and structure.

One of the most basic and useful relationships between elements that can be established is the most common order[2] in which we read a document, which is normally called reading order (see Figure 3.7).

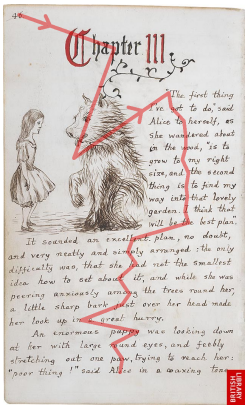


Figure 3.7: Example of a simple reading order. The red line follows the most common order in which each element in the document should be read.

[2]Notice that the way a document can be read is not unique, but normally there is a way that is more common than the others.

The Reading Order Determination is the definitive step that joins the data extracted by an ATR system and converts it into information. In this case, the layout structure h becomes more complex, because its elements must be ordered following the reading order, hence any independence assumption between elements must be carefully analyzed.

That is, h is restricted by the relationship $r \in l$, where r defines the most common reading order such that the probability that some element e' is read before other element e is greater than the opposite, this is:

$$p(r = 1 \mid e', e) \geq p(r = 0 \mid e', e) \quad \forall e', e \in h, e' \neq e \tag{3.3}$$

where $r = 1$ when e' must be ordered before e and $r = 0$ otherwise. Note that e' and e are not interchangeable in $p(r \mid e', e)$; that is e', e should not be seen as conventional conjunction of two individual conditions but as a single condition by itself.

Notice that this task can be applied to any of the previous tasks, where h can be a set of baselines (Section 3.3.3), regions of interest (3.3.4), pages (Section 3.3.6), or any other layout element. Also, it differs from general linking analysis because in the general case we can have several relationships between elements, but here we aim to extract only one of them.

To the best of our knowledge, we are the first ones to address the Reading Order Determination problem for handwritten documents specifically [QV21]. In contrast, research in the document Reading Order Determination task has a relatively long tradition for printed documents.

Lee et al. [Lee+02] proposed a system to translate an image of a printed document into a hyperdocument (in HTML format). While the success of the system is very dependent on the extraction of the structure of the input document, only a set of handcrafted rules are used to estimate the reading order. As a result, the system depends on specific domain knowledge.

A similar collection of rules were used in [Bre03] as a set of pairwise constraints, at the text-line level, to define a partial order between elements. Afterward, topological sorting is applied to extend this partial order to a total order of all elements in the document.

A different approach is given in [MCB08], where instead of defining the rules, they are automatically learned from training data using first-order logic theory. Then, a directed graph is built using all elements in the document as nodes, and the edges are defined by the learned rules. Finally, the reading order is extracted using a custom algorithm over the graph.

Recently, a new solution has been proposed for this problem as a byproduct of a more general DLA approach. In [NNC19] digitized newspapers are segmented into articles by means of the Viterbi decoding algorithm based on a 2D Markov Model. Although the advantage of an integrated approach is very clear, the size

of the state space grows exponentially with the number of elements which makes the method computationally expensive for DLA at text-line level or documents with many blocks.

Like in the previous case, in [PDM19] they focus on table analysis, but it has been foreseen that the reading order can be obtained as a byproduct of the process. The problem is modeled as an edge categorization problem over a graph, where the nodes represent text lines (baselines) and the edges the geometric relations between them. While the graph representation has many advantages for this task, it suffers from the complexity associated with the number of nodes and edges. In that case, they use a set of assumptions to make the problem tractable. A reduced graph is built with edges only in the neighborhood of each node instead of a fully connected graph. Those assumptions should be revised and updated for each kind of document.

In Chapter 6 we formalize the Reading Order Determination problem as a sorting problem, where the order-relation operator is probabilistically estimated and learned from examples. Then, we analyze its performance experimentally in Chapter 7.

3.3.6 Inter Page Analysis

Many times we can found collections of documents where an element of interest is spread over two or more pages. For example, a piece of news in a newspaper can begin on one page and end on another. This task aims to recognize those elements and establish the relationship between them so the information can be presented and processed in the correct way.

Formally, the input images cannot straightforwardly be assumed independent anymore, hence the complexity of the task grows significantly. For this reason, it is very common to carefully assume that, at first, all elements and images are independent and perform Page-level Analysis, then further methods are developed to address the inter-image elements.

For instance, in [Bos20; Pri+20] they split a collection of documents into "records", by detecting, at the image level, any possible *beginning*, *middle* and *end* of a "record" in a page, then all the data between the *beginning* of a "record" and the next *end* found is supposed to define a "record".

Although we present no direct contribution at this taxonomic level, it is straightforward to extend the formal methods developed in this dissertation to address problems at this level. We will formalize it in future work. That is, we want to obtain the elements e of the layout h, but restricted to all the collection $\mathcal{X}$:

$$\hat{h} = \arg\max_{h \in H} \prod_{e \in h} p(e|\mathcal{X}) \tag{3.4}$$

4 Document Text-line Analysis

Text-line analysis is one of the major sub-task in the DLA research area. Mainly because of its direct applicability to Automatic Text Recognition (ATR) systems such as HTR, KWS and OCR. Most SOTA ATR systems are able to recognize only one line of text at a time [GFG06; Pui18], which means that in order to recognize the text in a document, a DLA system must extract those text lines from the document first.

Text-line analysis is commonly divided into two main sub-tasks: text-line segmentation, which aims to obtain the placement of each text line in a page image, and text-line extraction (discussed in Section 4.3), which aims to split the input image into a set of smaller images that correspond to the segmented text lines.

In general, text-line segmentation can be handled in three main ways. First, the DLA system is designed to determine a polygon that best surrounds the text written down in a specific text line (see Figure 4.1a and Figure 4.1b). This kind of systems are very expensive to train and test since the manual labeling of the text lines is a cumbersome process and the border between two text lines could be fuzzy on handwritten documents. For instance, Figure 4.1a and Figure 4.1b are both valid segmentation options, however it is unclear which one is better.

In the second way, the DLA system is designed to extract only a sketchy polygon around the text line, while detailed detection of the polygon is handled to a second system, or —as in most of the modern cases—it is assumed that the ATR system is able to ignore the parts that are not strictly part of the text line (see Figure 4.1c). This approach has the advantage that the sketchy polygon can be defined by a few points (at least 4).

In the third way, the text-line segmentation problem is further simplified to detect only the baseline (the imaginary lines upon which the lines of text rest). Then, the text-line itself is extracted by another process (see Figure 4.1d). Nowadays this is the most common way to address the problem; because the labeling process is inexpensive. In fact, in most cases, a baseline can be defined by only two or three points.

In this work, we focus on baseline detection, because the process of manual labeling to create a ground-truth and further revision of the DLA is easier and cheaper. Also, it has been experimentally demonstrated that modern ATR systems are robust to noise [Rom+15], so a rough text line extracted from the baselines is

(a) (b) (c) (d)

Figure 4.1: Example of different ways to define the polygon that best surrounds the text line. a) a cumbersome detailed polygon, b) a detailed polygon, that is still cumbersome, c) an easy to define sketchy polygon, d) a simple polygon defined from the baselines.

enough for most of those systems (see Section 4.2). Nevertheless, for the sake of completeness, in Section 4.2 and Section 4.3 we refer the reader to key works on text-line segmentation and text-line extraction.

Baseline detection is still a difficult problem to solve. For this reason, we work over the following assumptions across this chapter:

- Page independence: we assume that each page $\mathbf{x} \in \mathcal{X}$ is independent of any other in the same set. This assumption allows us to avoid the computationally cumbersome process of analyzing all images together, instead, a single image is analyzed at a time. In general, this assumption is reasonable for the baseline detection problem due to the fact that no baseline extends to more than one page at a time.

- Element independence: we assume that each element $e \in h$ is independent of any other in the layout h. Although this is a hard assumption (e.g., the existence of a baseline depends on the fact that another one does not exist in the same place), it allows us to address the problem under limited computational resources.

In Section 7.3 we provide experimental evidence that supports that competitive results can be obtained under those assumptions.

4.1 Baseline Detection

As mentioned before, a baseline is the imaginary line upon which each line of text rests. It is commonly represented by a Piecewise Linear Curve (PLC), as shown in Figure 4.2.

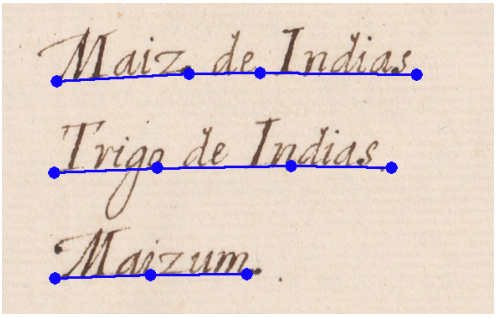


Figure 4.2: Example of a baseline represented by a PLC, where each blue point represents a vertex of it.

Although this representation is compact and convenient for human labeling, it is difficult to be handled by ANNs, because its dynamic and structured nature (variable number of vertexes by baseline and variable number of baselines per page) makes the training process computationally expensive or even unstable if only few training samples are available.

Keeping in mind that we want to model the posterior probability of the baselines in the layout given the input image ($P(h \mid X)$) employing an ANN, two different approaches are analyzed. The first one, which we call *Map-based* approach (Section 4.1.1), is based on the key idea of transforming the layout h of a document image into a simpler map, and then it is used to solve the problem instead of using h directly.

The second one is a *Direct* approach where h is modeled directly, but under the assumption that each part of h is independent. This approach is presented in Section 4.1.2.

4.1.1 Map-based Approach

Instead of using directly the PLC that defines a baseline, an intermediate representation can be used to train a probabilistic model (ANN, CRF, etc.). In this work, we

focus in ANN models due to their demonstrated versatility and capabilities.

In order to model $\mathrm{P}(h \mid X)$ using an ANN, a key idea is to transform the layout into a simpler map of *labels*. For instance, let $\mathbf{x} \in \mathbb{R}^d$ by a document image and $\mathcal{F} : H \to \mathbb{N}^{d'}$ a function that maps $h \in H$, the layout associated to $\mathbf{x}$, into a matrix of *labels*[1] $\mathbf{m} \in \mathbb{N}^{d'}$. This map $\mathbf{m}$ is a representation of the layout h that can be directly learned by an ANN.

If we assume that each point in the map $\mathbf{m}$, in the same position as the point $\mathbf{p} \in \mathbb{N}^2$ in $\mathbf{x}$, is independent of any other, we can write down the conditional probability of a map $\mathbf{m}$ giving an image $\mathbf{x}$:

$$\mathrm{P}(M = \mathbf{m} \mid X = \mathbf{x}) \overset{*}{=} \prod_{\mathbf{p} \in \mathbf{x}} \mathrm{P}(M_{\mathbf{p}} = m_{\mathbf{p}} \mid X = \mathbf{x}) \tag{4.1}$$

This model assigns a *label* to each pixel in the image. In the case of Baseline Detection problem, the *label* is 1 if the pixel belongs to the baseline representation or 0 otherwise.

A simple map $\mathcal{F}$ is defined by us in [Qui+18a; QTV19] and similarly by [Grü+19; ASK18] for each point $\mathbf{p} \in \mathbb{N}^2$ in $\mathbf{x}$ as defined in Equation (4.2) and depicted in Figure 4.3.

$$m_{\mathbf{p}} = \mathcal{F}_{\mathbf{p}}(h) = \begin{cases} 1 & \text{if } \exists\, e \mid d(\mathbf{p}, e) < \delta, e \in h \\ 0 & \textit{otherwise} \end{cases} \tag{4.2}$$

where $d(\mathbf{p}, e)$ is the minimal Euclidean distance between the point $\mathbf{p}$ and any point in the PLC defined by e, and δ is a hyperparameter defined for each dataset as the maximum value that generally leads to not overlaps in $\mathbf{m}$.

To define the layout h as a function of the distribution over the labeled pixels of the image, we use marginalization:

$$\begin{aligned} \mathrm{P}(h \mid X) = \\ \sum_{\mathbf{m}} \mathrm{P}(h, M = \mathbf{m} \mid X = \mathbf{x}) = \\ \sum_{\mathbf{m}} \mathrm{P}(M = \mathbf{m} \mid X = \mathbf{x})\, \mathrm{P}(h \mid M = \mathbf{m}, X = \mathbf{x}) = \\ \sum_{\mathbf{m}:\mathcal{F}^{-1}(\mathbf{m})=h} \mathrm{P}(M = \mathbf{m} \mid X = \mathbf{x}) \end{aligned} \tag{4.3}$$

where the last equality holds due to the fact that $\mathcal{F}^{-1}(\cdot)$ is a function (i.e., a deterministic process), then $\mathrm{P}(h \mid M = \mathbf{m}, X = \mathbf{x})$ is simply a Dirac delta function

[1] Here we use d' to denote the dimension of $\mathbf{m} \in$ because it is inherited from the dimension d of $\mathbf{x}$. Specifically, the width and height of $\mathbf{m}$ is equal to the width and height of $\mathbf{x}$, but the number of channels can differ based on the type of image.

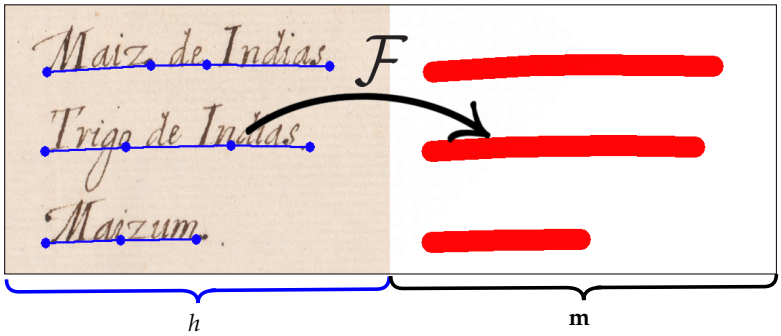


Figure 4.3: Example of a map $\mathbf{m}$ generated using the function $\mathcal{F}$ applied to a layout h. Here, h is defined as a set of baselines defined by PLCs (in blue).

whose value is 1 for the map that generates the reference set of baselines h and 0 for the rest.

Notice that this implementation is optimal if and only if $\mathcal{F}$ is bijective or at least an injective non-surjective function. In any other case, h cannot be unequivocally recovered from $\mathbf{m}$. For instance, in the case of Equation (4.2) the restriction $d(e, e') > 2\delta \; \forall e, e' \in h$ must hold and the number of vertexes in the reference baseline e must be minimal[2] in order to recover h.

As in Equation (3.2), the best layout $\hat{h}$ of some image $\mathbf{x}$ is defined by:

$$\hat{h} = \arg\max_{h \in H} P(h \mid \mathbf{x}) \tag{4.4}$$

where we are assuming that each image $\mathbf{x} \in \mathcal{X}$ is independent of the others. Hence, from Equation (4.3) we can obtain h in terms of the map $\mathbf{m}$ as:

$$\hat{h} = \arg\max_{h \in H} \sum_{\mathbf{m}:\mathcal{F}^{-1}(\mathbf{m})=h} \mathrm{P}(M = \mathbf{m} \mid X = \mathbf{x}) \tag{4.5}$$

and from Equation (4.1):

$$\hat{h} = \arg\max_{h \in H} \sum_{\mathbf{m}:\mathcal{F}^{-1}(\mathbf{m})=h} \prod_{\mathbf{p} \in \mathbf{x}} \mathrm{P}(M_{\mathbf{p}} = m_{\mathbf{p}} \mid X = \mathbf{x}) \tag{4.6}$$

4.1.1.1 Training Process

Once we have a way to efficiently compute the posterior probability of the baselines given the image (Equation (4.3)), we will follow the training criteria explained in

[2] A minimal PLC is such that, in addition to the start and end vertex, a new vertex is added only if there is a change in the direction of the line.

Section 2.3.1. Specifically, from Equation (2.9) and Equation (4.3) the *cross entropy* cost function takes the form:

$$J(\boldsymbol{\theta}) = \frac{-1}{N}\sum_{n=1}^{N}\frac{1}{d'}\sum_{\mathbf{p}\in\mathbf{x}_n} m_{n,\mathbf{p}}\log(\Phi_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta})) + (1-m_{n,\mathbf{p}})\log(1-\Phi_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta})) \quad (4.7)$$

where d' is the size of $\mathbf{m}$, $m_{n,\mathbf{p}} \in \{0,1\}$ is the map label of the sample n at point $\mathbf{p}$, and $\Phi_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta}) \in \boldsymbol{\Phi}, \boldsymbol{\Phi} : \mathbb{R}^d \to (0,1)^{d'}$ is the output of the ANN at point $\mathbf{p}$.

Furthermore, gradient-descent algorithms are often used (see Section 2.3.1) to (iteratively) minimize the cost function. Since all components in ANNs and CNNs are differentiable[3], we can compute the gradient of the cost function with respect to the network parameters. Therefore, back-propagation is used to efficiently compute the gradients and update the parameters.

4.1.1.2 Inference

Once the $\mathrm{P}(\mathbf{m} \mid \mathbf{x};\boldsymbol{\theta})$ distribution is learned by our model, one can now obtain a prediction of the layout of any new previously unseen image.

From Equation (4.6), the best layout $\hat{h}$ is the one that maximizes the sum of the probability over all the maps that can generate that layout. However, under the assumption that $\mathcal{F}$ is bijective or at least an injective non-surjective function, the inverse function $\mathcal{F}^{-1}(\mathbf{m})$ is unique. Hence, the most probable layout $\overset{*}{h}$ is defined by the most probable map $\mathbf{m}$. As DT states (see Section 2.2), the most probable map $\overset{*}{\mathbf{m}}$, given a new input image $\mathbf{x}$, can be obtained by taking the most probable value for each point $\mathbf{p}$ as:

$$\overset{*}{m}_{\mathbf{p}} = \begin{cases} 1 & \text{if} \quad \Phi_{\mathbf{p}}(\mathbf{x}) > 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (4.8)$$

Finally, the most probable document layout can be retrieved from the most probable map given by the model as:

$$\overset{*}{h} = \mathcal{F}^{-1}(\overset{*}{\mathbf{m}}) \quad (4.9)$$

where $\mathcal{F}^{-1}(\cdot)$ is the inverse function of Equation (4.2). Notice that the inverse of Equation (4.2) is equivalent to search for all points $\mathbf{p}$ that belongs to an element $e \in h$. That is, $\mathbf{p} \in e$ if all the elements if $\mathbf{m}$ inside a circle of radius δ centered on $\mathbf{p}$ are equal to 1 (i.e., $t_{\mathbf{p}} = 1$ (Equation (4.10))), and $\mathbf{p}$ belongs to the connected components neighborhood of e.

[3] Some activation functions could be not completely differentiable, but it is empirically proven that it has no adverse impact in the model. See Section 2.3 for details.

$$t_{\mathbf{p}} = \begin{cases} 1 & \text{if} \quad m_{\mathbf{j}} = 1 \forall \mathbf{j} \mid d(\mathbf{p}, \mathbf{j}) \leq \delta \\ 0 & \text{otherwise} \end{cases} \tag{4.10}$$

As explained in Section 3.2, the layout of a document is not unique. Consequently, different layouts can be interpreted by a human user to be correct, while from a formal point of view only the one equal to the reference is correct (i.e., $\hat{h} = \overset{*}{h}$). Also, in most cases, the baselines are not minimal PLCs. Furthermore, formally $\mathcal{F}^{-1}(\cdot)$ is not flexible enough to handle the uncertainty characteristic of a probabilistic model like an ANN. For instance, the inverse is very sensitive to false-negative errors,e.g., if in some hypothesis $\overset{*}{\mathbf{m}}$ a point $\mathbf{k}$ in the neighborhood of radio δ of $\mathbf{p} \in e$ is miss-classified, then $m_{\mathbf{k}} = 0 \implies t_{\mathbf{p}} = 0$, and the point $\mathbf{p}$ will not be added as part of the element e as expected.

For those reasons, although it is possible, searching for the formally correct layout it is most of the time cumbersome and unnecessary. Instead, several approximations could lead to solutions that are feasible and also correct given the fuzziness of the problem.

Based on the fact that $\overset{*}{\mathbf{m}}$ can be interpreted as a binary image of the same size of $\mathbf{x}$, in [Qui+18a] we presented a solution based on a well-known computer vision algorithm used to compute the CCs in a binary image followed by the algorithm defined in [PV94] to optimally reduce the number of vertexes of a PLC.

This algorithm is presented in Algorithm 1, where the ConnectedComponents procedure is implemented using the algorithm developed by Suzuki et al. [Suz+85], and the GenBaseline procedure, presented in Algorithm 2, uses each of the extracted CC as a guide to search for the baseline in the image $\mathbf{x}$. The main steps of this procedure are depicted in Figure 4.4. In addition, it is important to notice that the ReducePolygon procedure is very robust to outliers (see [PV94] for details), which makes the procedure helpful not only to reduce the number of vertexes, but to remove outliers.

Algorithm 1 Approximation of $\mathcal{F}^{-1}(\cdot)$ based on CC for the baseline detection problem.

Require: $\mathbf{m}$ is a binary map, $\mathbf{x}$ is an image, k the number of vertexes of each baseline

```
procedure GenLayout(m,x)
    h ← {∅}                                          ▷The layout
    Γ ← ConnectedComponents(m)                       ▷ One baseline per CC will be generated
    for all ϱ ∈ Γ do
        h ← h ∪ GenBaseline(ϱ, x, k)                 ▷From Algorithm 2
    end for
    return h
end procedure
```

Algorithm 2 Baseline detection in a restricted region

Require: $\mathbf{x}$ is an image, ϱ is a contour, k is the number of vertexes of the output PLC

procedure GenBaseline(ϱ, $\mathbf{x}$, k)
  $\mathbf{I} \leftarrow \text{Crop}(\mathbf{x}, \varrho)$ ▷Crop of the image inside ϱ.
  $\mathbf{Y} \leftarrow \text{Otsu}(\mathbf{I})$ ▷Binarize $\mathbf{I}$ using Otsu algorithm.
  $\alpha \leftarrow \text{MainDirection}(\mathbf{Y})$ ▷Get $\mathbf{Y}$'s main direction. $\alpha = 0$ if horizontal or $\alpha = 1$ if vertical).
  $r, c \leftarrow \text{size}(\mathbf{Y})$ ▷Number of rows r and columns c of $\mathbf{Y}$
  $l \leftarrow 0$
  if $\alpha == 0$ **then**
    for $j = 0, j < c, j++$ **do**
      for $i = r - 1, i \geq 0, i--$ **do**
        if $Y_{i,j} == 1$ **then**
          $e[l++] = (i, j)$ ▷Add the point i, j to the bulk PLC e.
          $l \leftarrow l + 1$
          endloop
        end if
      end for
    end for
  else
    for $i = 0, i < r, i++$ **do**
      for $j = 0, j < c, j--$ **do**
        if $Y_{i,j} == 1$ **then**
          $e[l++] = (i, j)$ ▷Add the point i, j to the bulk PLC e.
          $l \leftarrow l + 1$
          endloop
        end if
      end for
    end for
  end if
  if $\text{len}(e) > k$ **then**
    $e \leftarrow \text{ReducePolygon}(e, k)$ ▷Reduce the number of vertexes to k using Perez et al. [PV94] algorithm.
  end if
  return e
end procedure

The main advantage of this algorithm, along with its simplicity, is that it uses the input image to be more robust against false-negative errors in $\overset{*}{\mathbf{m}}$. In contrast, it is sensitive to false-positive errors. For instance, it cannot recover the correct layout in cases where the expected baselines in $\overset{*}{\mathbf{m}}$ overlap or at least touch each other.

In practice, this disadvantage can be mitigated by applying a morphological erosion to the hypothesis $\overset{*}{\mathbf{m}}$.

4.1.2 Direct Approach

The previous approach cannot handle overlaps naturally. For instance, in some cases, the text lines on a table are very close but separated by the cell boundary. In those

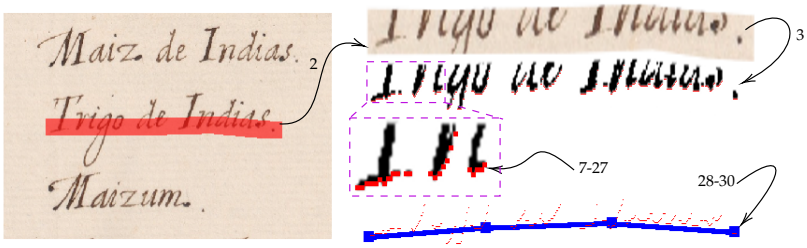


Figure 4.4: Main steps followed by Algorithm 2, where the numbers at the side of each arrow are the corresponding line number in Algorithm 2. First the image defined by the contour ϱ (in red), predicted over the text line "Trigo de Indias", is cropped and assigned to $\mathbf{I}$. Then, $\mathbf{I}$ is binarized (line 3 in Algorithm 2). Next, only the "last" black pixels of each column are added to the baseline (red points, lines 7 to 27). Finally, the PLC defined by those points is reduced to k vertexes (blue line).

cases, it is impossible to select a hyperparameter δ big enough to produce a useful map but small enough to avoid overlaps.

Instead, the called *Direct* approach can handle those overlaps naturally, where $\mathrm{P}(h \mid X = \mathbf{x})$ is modeled under the assumption that each layout element (baseline) is independent of any other:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{*}{=} \prod_{e \in h} \mathrm{P}(e \mid X = \mathbf{x}) \tag{4.11}$$

where each element e is composed of a bounding box $\mathbf{b}^e \in \mathbb{R}^4$, with a class $c^e = \{0, 1\}$ and a map $\mathbf{m}^e = 1 \iff d(\mathbf{p}, e) < \delta, \mathbf{p} \sqsubseteq \mathbf{b}^e$. Here we abuse the notation and use $\mathbf{p} \sqsubseteq \mathbf{b}$ to define that a point $\mathbf{p}$ is inside of the bounding box $\mathbf{b}$. This map is indeed the same defined before in Equation (4.2), but restricted to the bounding box of the element e instead of the full image. Notice that c^e and $\mathbf{m}^e$ are enough to fully describe the layout element e, but we also define $\mathbf{b}^e$ so it could play an important role (as detailed in Section 4.1.2.1) in training our probabilistic model.

Then Equation (4.11) can be re-written in terms of the joint probability of the parts of each layout element as:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{*}{=} \prod_{e \in h} \mathrm{P}(\mathbf{b}^e, c^e, \mathbf{m}^e \mid X = \mathbf{x}) \tag{4.12}$$

Furthermore, if we assume that each part of an element is independent of the other parts, the Equation (4.12) can be simplified to:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{*}{=} \prod_{e \in h} \mathrm{P}(\mathbf{b}^e \mid X = \mathbf{x})\, \mathrm{P}(c^e \mid X = \mathbf{x})\, \mathrm{P}(\mathbf{m}^e \mid X = \mathbf{x}) \tag{4.13}$$

Therefore, $\mathrm{P}(h \mid X = \mathbf{x})$ can be modeled independently for each part of the baseline e as a multi-task problem.

4.1.2.1 Training Process

Now that the Baseline Detection problem is formalized as a multi-task problem, we handle it using an ANN, as we did in [QV21]. We define a three tasks network $\boldsymbol{\Phi}(\mathbf{x}, \boldsymbol{\theta})$, which consists of one regression task (bounding box estimator, $\boldsymbol{\Phi}_{\mathbf{b}}(\cdot)$) and two classification tasks (a class estimator $\boldsymbol{\Phi}_{c}(\cdot)$, and a map estimator $\boldsymbol{\Phi}_{\mathbf{m}}(\cdot)$):

$$\boldsymbol{\Phi}(\mathbf{x}, \boldsymbol{\theta}) = \{\boldsymbol{\Phi}_{\mathbf{b}}(\mathbf{x}, \boldsymbol{\theta}_{\mathbf{b}}, \boldsymbol{\theta}_s), \Phi_{c}(\mathbf{x}, \boldsymbol{\theta}_{c}, \boldsymbol{\theta}_s), \boldsymbol{\Phi}_{\mathbf{m}}(\mathbf{x}, \boldsymbol{\theta}_{\mathbf{m}}, \boldsymbol{\theta}_s)\} \tag{4.14}$$

where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_s, \boldsymbol{\theta}_{\mathbf{b}}, \boldsymbol{\theta}_{c}, \boldsymbol{\theta}_{\mathbf{m}}\}$, $\boldsymbol{\theta}_s$ is a set of shared parameters, while the rest are local parameters for each task.

Specifically, from Equation (4.13) we can use SGD optimizer and a cost function composed of the weighted combination of three cost functions (one for each task). Namely, binary cross entropy cost ($J_{\mathbf{c}}$) for c^e task, a $J_{\mathbf{b}}$ cost based on the L1 loss function for the bounding box task and binary cross entropy cost ($J_{\mathbf{m}}$) for the mask prediction task.

Then, the total cost between the posterior probability of each part of the layout elements and its correspondent approximations:

- $P(\mathbf{b}^e \mid \mathbf{x}; \boldsymbol{\theta}) \stackrel{\text{def}}{=} \boldsymbol{\Phi}_{\mathbf{b}^e}(\mathbf{x}, \boldsymbol{\theta})$, $\boldsymbol{\Phi}_{\mathbf{b}^e} : \mathbb{R}^d \to \mathbb{R}^4$
- $P(c^e \mid \mathbf{x}; \boldsymbol{\theta}) \stackrel{\text{def}}{=} \Phi_{c^e}(\mathbf{x}, \boldsymbol{\theta})$, $\Phi_{c^e} : \mathbb{R}^d \to [0, 1]$
- $P(\mathbf{m}^e \mid \mathbf{x}; \boldsymbol{\theta}) \stackrel{\text{def}}{=} \boldsymbol{\Phi}_{\mathbf{m}^e}(\mathbf{x}, \boldsymbol{\theta})$, $\boldsymbol{\Phi}_{\mathbf{m}^e} : \mathbb{R}^d \to [0, 1]^{d'_{\mathbf{m}^e}}$, where $d'_{\mathbf{m}^e}$ is the size of the map $\mathbf{m}^e$.

is defined as:

$$\begin{aligned}
J(\boldsymbol{\theta}) &= \lambda_{\mathbf{b}} J_{\mathbf{b}} + \lambda_{c} J_{c} + \lambda_{\mathbf{m}} J_{\mathbf{m}} \\
&= \frac{1}{N} \sum_{n=1}^{N} \Bigg(\lambda_{\mathbf{b}} \frac{1}{|h_n|} \sum_{e \in h_n} \| \mathbf{b}^e - \boldsymbol{\Phi}_{\mathbf{b}^e}(\mathbf{x}_n) \| \\
&\quad + \lambda_{c} \frac{-1}{|h_n|} \sum_{e \in h_n} c^e \log(\Phi_{c^e}(\mathbf{x}_n)) + (1 - c^e) \log(1 - \Phi_{c^e}(\mathbf{x}_n)) \\
&\quad + \lambda_{\mathbf{m}} \frac{-1}{|h_n|} \sum_{e \in h_n} \frac{1}{d'_{\mathbf{m}^e}} \sum_{k=1}^{d'_{\mathbf{m}^e}} m_k^e \log(\Phi_{\mathbf{m}^e,k}(\mathbf{x}_n)) + (1 - m_k^e) \log(1 - \Phi_{\mathbf{m}^e,k}(\mathbf{x}_n)) \Bigg)
\end{aligned} \tag{4.15}$$

where $\Phi_{*^e}(\cdot)$ implies that the model should generate hypotheses aligned to the elements in the reference h. Normally, a Region Proposal Generator (RPG) is used to generate a set of RoIs. Those RoIs are assigned to a reference element e if $\text{IoU}(\mathbf{b}^e, \mathbf{b}^{\text{RoI}}) > 0.5$ (see Section 2.3.3)

In order to optimize the RoIs generated by the RPG, different methods can be used. In this work, we use an RPN generator (see Section 2.3.3) to obtain the optimal number and placement of the RoIs. As mentioned in Section 2.3.3, a RPN is a kind of ANN, which is differentiable. Therefore, we can compute the gradient of the loss function of the proposed anchors (RoIs) with respect to the network parameters, and include this cost function (J_{RPN}) in the total weighted cost defined in Equation (4.15):

$$\mathcal{L}(h, \mathbf{x}) = \lambda_{\mathbf{b}} J_{\mathbf{b}} + \lambda_c J_c + \lambda_{\mathbf{m}} J_{\mathbf{m}} + \lambda_{\text{RPN}} J_{\text{RPN}} \tag{4.16}$$

4.1.2.2 Inference

Similarly to the *Map-based* approach, once the posterior distribution of each part of the layout elements is learned by our model, one can obtain the most probable layout $\overset{*}{h}$ given a new input image $\mathbf{x}$.

Let $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_n\}$ be the set of RoIs generated by an RPN. For each one of them we obtain its bounding box $\overset{*}{\mathbf{b}}^{\beta}$ and its class $\overset{*}{c}^{\beta}$ from Equation (4.17) and Equation (4.18) respectively.

$$\overset{*}{\mathbf{b}}^{\beta} = \boldsymbol{\Phi}_{\mathbf{b}^{\beta}}(\mathbf{x}) \tag{4.17}$$

$$\overset{*}{c}^{\beta} = \underset{i \in \{0,1\}}{\arg\max}\, \Phi_{c^{\beta},i}(\mathbf{x}) \tag{4.18}$$

Similarly to the *Map-based* approach, one can obtain the vertexes of the PLC using the GenBaseline$(\mathbf{x}, \varrho, k)$ procedure (see Algorithm 2), where ϱ is the perimeter of the most probable map $\overset{*}{\mathbf{m}}^{\beta}$ obtained by Equation (4.19).

$$\overset{*}{\mathbf{m}}^{\beta}_{\mathbf{p}} = \underset{i \in \{0,1\}}{\arg\max}\, \Phi_{\mathbf{m}^{\beta},\mathbf{p},i}(\mathbf{x}) \tag{4.19}$$

Finally, the most probable document layout can be retrieved from the most probable set of RoIs using Equation (4.20) as the subset of $\mathcal{B}$ which has been classified to belong to class 1 (i.e., baseline).

$$\overset{*}{h} = \{\beta \in \mathcal{B} \mid \overset{*}{c}^{\beta} = 1\} \tag{4.20}$$

4.2 Text-line Segmentation

Ultimately, the baseline detection problem main applicability is to provide a set of delimited text lines to be processed by an ATR system. To that end, we need to

obtain the polygon $\rho^e \in \mathbb{R}^{\geq 4}$ for each baseline e that *best surrounds* the text around the baseline.

Given the fact that there are several good methods developed to obtain this polygon from the baselines, we believe the problem is mature enough to be considered out of the scope of this dissertation. However, in this section, we refer the reader to some key works and methods to obtain the desired polygon.

It is not out of controversy how *best* is defined. For instance, one can define that a polygon is *best* if it contains all the pixels in the input image that belong to the text line and none of the background pixels. This definition will lead to a cumbersome labeling process which —as empirically demonstrated by [Rom+15]—is needless for modern ATR systems (see Figure 4.1a). Besides, some works like [MLF14; Bos+18] have developed methods to obtain the polygon that best fits that definition.

On the other hand, in accordance with the empirical results obtained by [Rom+15], the *best* polygon is allowed to enclose background pixels in order to keep it as simple as possible. The level of complexity of the polygon depends on the complexity of the document, but since some background is allowed the number of vertexes needed are much lower (see Figure 4.1c). For instance, in [BTV12; Moy+15] the polygon is defined by a few vertexes (normally less than 20), which makes it easier to be edited by the user if necessary.

A minimal case can be carried out when the text is very homogeneous. A simple parallel offset over the PLC is enough to segment the text line (see Figure 4.1d). This polygon can be generated from e following Algorithm 3, where $\boldsymbol{\kappa}_0 \in \mathbb{N}^2$ is an offset to the left (top) side of the baseline and $\boldsymbol{\kappa}_1 \in \mathbb{N}^2$ is an offset to the right (bottom) side.

Algorithm 3 Simple polygon generator around a baseline.

Require: e is a PLC, $\boldsymbol{\kappa}_0$ is the top offset, $\boldsymbol{\kappa}_1$ is the bottom offset.

procedure GenPolygon(e,$\boldsymbol{\kappa}_0$, $\boldsymbol{\kappa}_1$)
  $n \leftarrow \text{len}(e)$ ▷Get the number of vertexes in e
  $e' \leftarrow [(0,0)]^{2n}$ ▷New polygon e' will have $2n$ vertexes.
  for $1 \leq i \leq n$ **do**
    $e'_i \leftarrow e_i + \boldsymbol{\kappa}_0$ ▷Upper part of the polygon.
    $e'_{n+i} \leftarrow e_{n+1-i} - \boldsymbol{\kappa}_1$ ▷Bottom of the polygon.
  end for
  return e'
end procedure

Generally, $\boldsymbol{\kappa}_0$ is defined as a vector of magnitude 1.5 times the average x-height[4] and direction pointing to the upper part of the text, and $\boldsymbol{\kappa}_1$ is another vector with magnitude 0.75 times the average x-height, and pointing to the bottom of the text.

[4] X-height refers to the height of the lowercase symbols.

(a) (b) (c) (d)

Figure 4.5: Example of different ways to extract the text line "Maizum" given a segmentation. a) a segmentation of few lines where the text line is extracted as: b) the straight bounding rectangle, c) the rotated rectangle with the minimum area, and d) the masked rotated rectangle with the minimum area.

4.3 Text-line Extraction

Once text lines are detected and segmented, they are commonly used by some ATR systems to recognize the data recorded on them. But, to the best of our knowledge, all modern ATR systems require that the image of each text line must be a rectangle. Hence, the polygon ρ that represents a segmented text line should be converted into a simple rectangle $\gamma \in \mathbb{R}^4$ using, for instance, one of the following approaches:

- γ is the straight bounding rectangle of ρ. This rectangle is easily computed, but in cases where e is rotated, γ may include data from other text lines. For instance, in Figure 4.5b a bit of the letter "g" from the previous text line is included in the rectangle.
- γ is the rotated rectangle with the minimum area that surrounds ρ (see Figure 4.5c).
- γ is the rotated or straight bounding rectangle of ρ, but the pixels that are placed outside the polygon ρ and inside γ are set to a fixed value (normally zero, see Figure 4.5d).
- ρ is transformed before converted into γ. In this approach, ρ is pre-processed in several ways in order to improve the likelihood for an ATR system to recognize the correct data. For example, it is common to apply warping correction techniques [Pil01; ZT03], skew correction [GPC97] and slope/slant correction [Ber+07; DC12].

5 *Document Page Level Analysis*

Most of the guides on writing advise that it is essential to divide the text into structured pieces that guide the reader towards a good understanding of the information to be conveyed. For instance, for a news article, it is recommended to draw the reader's attention in the first paragraph, while stating the story's who, what, when, where, and why. Then, details are given in the following paragraphs, and finally, in the last paragraph, the writer should end with a quote or a catchy phrase that helps the reader to evoke and remember what the article was about easily. Equally important is adding visual clues (graphs, photos, drawings, etc.) that compliment the message and help the reader to understand it.

In the previous chapter, we proposed some ideas for obtaining the lines of text present in a document so an ATR system can be used to obtain the transcription of those. However, it is clear that the raw transcription obtained from those lines is not enough to fully recover the information in a document. For instance, we should consider how those lines are grouped (e.g., paragraphs), the logical meaning of those groups (e.g., headers, marginalia) and, certainly, if there is any visual clue that complements the message.

To that end, the methods discussed in this chapter aims to split the page into a set of elements of interest, taking into account not only the text related elements (paragraphs, page-number, marginalia, etc.) but any other kind of region like images, drawings, stamps, or any other relevant structure present in the page. This problem is commonly referred to in the scientific community as *Region Segmentation*.

5.1 Region Segmentation

It is not uncommon to find some works in the literature that understand Region Segmentation as the problem of geometrically splitting a document into a set of regions of interest (i.e., geometric layout). Although in this dissertation we prefer to follow a broader definition of the problem, in which not only geometric layout is performed, but to some extent logical layout as well (in the form of region labeling).

In general, the problem is defined as follows: *Given an input document* $\mathbf{x}$*, obtain the set of layout regions that better explain the structure of the document, along with the corresponding region label*.

With this in mind, the layout h of $\mathbf{x}$ becomes not only a set of polygons but a set of polygons with a class attribute $h = \{e_1, e_2, \ldots, e_K\}$ and $e_k = \{\boldsymbol{\rho}, c\}$, where $\boldsymbol{\rho} \in \mathbb{N}^v$ is the shape definition (i.e., a polygon) and $c \in \mathcal{Y}$ is its class label from the set $\mathcal{Y}$ that contains the labels assigned to each region type.

Like the PLC used to represent a baseline, a polygon is very useful for representing a region of interest. It is very compact and convenient for human labeling as we can define any region with different levels of complexity (e.g., from a minimum three vertex polygon or the familiar four vertex rectangle to a very complex shape). However, it shares the same drawbacks for training to be handled by an ANN (see Section 4.1).

Nevertheless, the same formulation presented in Chapter 4 for the Baseline Detection problem can be extended to address the Region Segmentation problem. Similarly to the previous chapter, here we develop the *Map-based* approach (Section 5.1.1) and the *Direct* approach (Section 5.1.2) to help us to model the posterior probability of the regions of interest in the layout ($P(h \mid X)$) using an ANN. Then, in Section 7.4, we evaluate the proposed methods experimentally.

Furthermore, as the reader may notice, it is straightforward to extend the very same formulation to handle both Baseline Detection and Region Segmentation problems in an integrated way. Indeed, we do so in Section 5.2 and evaluate it in Section 7.5.

5.1.1 Map-based Approach

Similarly to Section 4.1.1, the main idea is to transform the regions $e \in h$ into a simple map of labels. For instance, let $\mathbf{x} \in \mathbb{R}^d$ be a document image and $\mathfrak{F} : H \to \mathbb{N}^{d'}$ a function that maps $h \in H$, the layout associated to $\mathbf{x}$, into a matrix of labels $\boldsymbol{\xi} \in \mathbb{N}^{d'}$. This map $\boldsymbol{\xi}$ is a representation of the layout h that an ANN can learn directly.

Again, if we assume that each point in the map $\boldsymbol{\xi}$, in the same position as the point $\mathbf{p} \in \mathbb{N}^2$ in $\mathbf{x}$, is independent of any other. Then Equation (4.1) holds for $\boldsymbol{\xi}$ in the same way it does for $\mathbf{m}$:

$$\mathrm{P}(M = \boldsymbol{\xi} \mid X = \mathbf{x}) \overset{*}{=} \prod_{\mathbf{p} \in \mathbf{x}} \mathrm{P}(\boldsymbol{\Xi}_{\mathbf{p}} = \xi_{\mathbf{p}} \mid X = \mathbf{x}) \tag{5.1}$$

This model assigns a *label* to each pixel in the image. In the case of the Region Segmentation task, the *label* is conveniently defined as 0 if the pixel does not belong to any region in the image (i.e., the background) or as $c \in \mathcal{Y}$ if the pixel belongs to a region with class label c.

We define a simple map $\mathfrak{F}$ in [Qui+18a; QTV19], and similarly by [ASK18], for each point $\mathbf{p} \in \mathbb{N}^2$ in $\mathbf{x}$ as defined in Equation (5.2) and depicted in Figure 5.1.

$$\xi_{\mathbf{p}} = \mathfrak{F}_{\mathbf{p}}(h) = \begin{cases} c^e & \text{if } \mathbf{p} \sqsubseteq \rho^e, e \in h \\ 0 & otherwhise \end{cases} \tag{5.2}$$

where we abuse the notation and use $\mathbf{p} \sqsubseteq \rho^e$ to denote that a point $\mathbf{p}$ is inside the polygon that defines the shape of the element e, and $c^e \in \mathcal{Y}$ is the class of the element e.

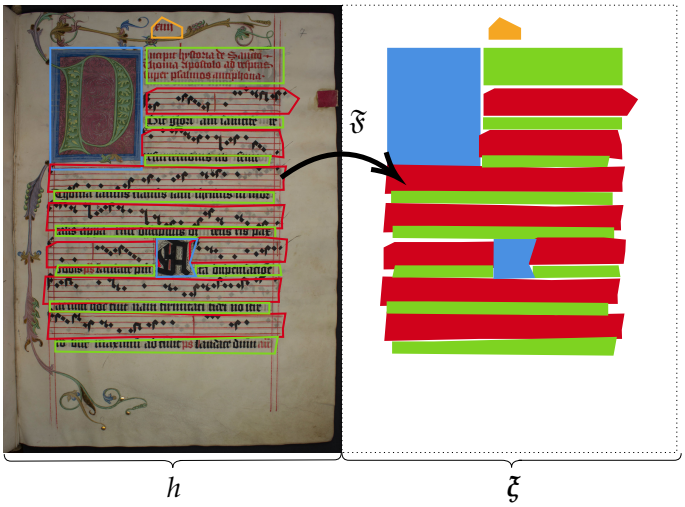


Figure 5.1: Example of a map function $\mathfrak{F}$ applied to a layout h. Notice that $\mathfrak{F}$ do not avoid overlaps between different regions.

Furthermore, we define h in terms of ξ using marginalization, as we did in Equation (4.3):

$$\mathrm{P}(h \mid X) = \sum_{\xi : \mathfrak{F}^{-1}(\xi) = h} \mathrm{P}(\Xi = \xi \mid X = \mathbf{x}) \tag{5.3}$$

where this equality holds due to the fact that $\mathfrak{F}^{-1}(\cdot)$ is a function (i.e., a deterministic process), then $\mathrm{P}(h \mid \Xi = \xi, X = \mathbf{x})$ is simply a Dirac delta function whose value is 1 for the map that generates the reference set of regions h and 0 for the rest.

Notice that this implementation is optimal if and only if $\mathfrak{F}$ is bijective or at least an injective non-surjective function. In any other case, h cannot be unequivocally recovered from ξ. For instance, in the case of Equation (5.2) it is clear that overlaps between two or more elements will lead to a case where not enough information

remains to reverse the process and obtain unequivocally the region used to generate some values in the map. Hence, any element e must not overlap or touch any other region in order to fully recover h.

Please also note that, it could be an important limitation of the method, as in many cases it is tough to manually label the document without any overlap, especially in the boundary between different regions (see Figure 5.1). However, due to the fuzziness of the problem (see Chapter 3), this limitation is negligible in many practical cases.

Moreover, as in Equation (3.2), the best layout $\hat{h}$ of some image $\mathbf{x}$ is defined by:

$$\hat{h} = \underset{h \in H}{\arg\max}\, P(h \mid \mathbf{x}) \tag{5.4}$$

where we are assuming that each image $\mathbf{x} \in \mathcal{X}$ is independent of the others. Hence, from Equation (5.3) we can obtain h in terms of the map $\boldsymbol{\xi}$ as:

$$\hat{h} = \underset{h \in H}{\arg\max} \sum_{\boldsymbol{\xi}:\mathfrak{F}^{-1}(\boldsymbol{\xi})=h} \mathrm{P}(\Xi = \boldsymbol{\xi} \mid X = \mathbf{x}) \tag{5.5}$$

and from Equation (5.1):

$$\hat{h} = \underset{h \in H}{\arg\max} \sum_{\boldsymbol{\xi}:\mathfrak{F}^{-1}(\boldsymbol{\xi})=h} \prod_{\mathbf{p} \in \boldsymbol{\xi}} \mathrm{P}(\Xi_{\mathbf{p}} = \xi_{\mathbf{p}} \mid X = \mathbf{x}) \tag{5.6}$$

5.1.1.1 Training Process

Following the same formulation as in the Baseline Detection problem (Section 4.1.1.1), the posterior probability of h given an input image $\mathbf{x}$ is estimated by means of an ANN. However, as a document could have many different types of regions, the architecture and cost function must be redesigned as a multinomial classification problem.

Specifically, from Equation (2.11) and Equation (5.3) we minimize the *cross entropy* cost funtion in the form:

$$J(\boldsymbol{\theta}) = \frac{-1}{N} \sum_{n=1}^{N} \frac{1}{d'} \sum_{\mathbf{p} \in \mathbf{x}_n} \boldsymbol{\xi}'_{n,\mathbf{p}} \log(\boldsymbol{\Phi}_{\mathbf{p}}(\mathbf{x}_n, \boldsymbol{\theta})) \tag{5.7}$$

where $\boldsymbol{\xi}'_{n,\mathbf{p}} \in \{0,1\}^{|\mathcal{Y}|+1}$ is the one-hot representation of the target value $\xi_{n,\mathbf{p}}$ of the sample n, $\boldsymbol{\Phi}_{\mathbf{p}}(\cdot), \boldsymbol{\Phi}_{\mathbf{p}} : \mathbb{R}^d \to [0,1]^{|\mathcal{Y}|+1}$ is the output of the ANN (softmax function) at point $\mathbf{p}$, and in general $\boldsymbol{\Phi} : \mathbb{R}^d \to [0,1]^{d' \times |\mathcal{Y}|+1}$.

5.1.1.2 Inference

Once $P(\boldsymbol{\xi} \mid \mathbf{x}; \boldsymbol{\theta})$ distribution is learned by our model, and assuming that $\mathfrak{F}$ is bijective or at least an injective non-surjective function, one can now predict the layout of any previously unseen image.

Following Equation (5.6), the best layout $\hat{h}$ is the one that maximizes the sum of the probability over all the maps that can generate that layout. However, as we assume that $\mathfrak{F}$ is bijective or at least an injective non-surjective function, the inverse function $\mathfrak{F}^{-1}(\boldsymbol{\xi})$ is unique. Hence, the most probable layout $\overset{*}{h}$ is directly defined by the most probable map $\boldsymbol{\xi}$. As DT states (see Section 2.2), the most probable map $\overset{*}{\boldsymbol{\xi}}$, given a new input image $\mathbf{x}$, can be obtained by taking the most probable value for each point $\mathbf{p}$ as:

$$\overset{*}{\xi}_{\mathbf{p}} = \underset{i \in \{c \in \mathcal{Y}, 0\}}{\arg\max}\, \Phi_{\mathbf{p},i}(\mathbf{x}) \tag{5.8}$$

where $i = 0$ represents the background, and $\Phi_{\mathbf{p},i}(\cdot)$ is the estimated probability of the point $\mathbf{p}$ to belong to the class i.

Finally, the most probable document layout can be retrieved from the most probable map given by the model as:

$$\overset{*}{h} = \mathfrak{F}^{-1}(\overset{*}{\boldsymbol{\xi}}) \tag{5.9}$$

where $\mathfrak{F}^{-1}(\cdot)$ is the inverse of Equation (5.2), which is equivalent to search for all CCs in $\overset{*}{\boldsymbol{\xi}}$. For instance, by means of a ConnectedComponents procedure implemented using the algorithm presented by Suzuki et al. [Suz+85], then:

$$\overset{*}{h} \leftarrow \textsc{ConnectedComponents}(\overset{*}{\boldsymbol{\xi}}) \tag{5.10}$$

The main advantage of this algorithm is its simplicity. In contrast, it is sensitive to false-positive errors. For instance, it cannot recover the correct layout in cases where the expected layout regions in $\boldsymbol{\xi}$ overlap or at least touch each other, as they will be merged into a single region by the ConnectedComponents procedure. In practice, this disadvantage can be mitigated by applying morphological operations to the hypothesis $\overset{*}{\boldsymbol{\xi}}$.

5.1.2 Direct Approach

Similarly to Section 4.1.2, assuming that each layout region is independent of any other in h, the *Direct* approach can be applied to layout regions as well as it is to baselines:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{\text{def}}{=} \prod_{e \in h} P(e \mid X = \mathbf{x}) \tag{5.11}$$

where, in this case, e is a layout region described using the bounding box $\mathbf{b}^e \in \mathbb{R}^4$ that best surrounds $\boldsymbol{\rho}^e$, with a class $c^e \in \mathcal{Y}$ and a map $\boldsymbol{\xi}^e = 1 \iff \mathbf{p} \sqsubseteq \boldsymbol{\rho}^e$. Again, we abuse the notation and use $\mathbf{p} \sqsubseteq \boldsymbol{\rho}^e$ to denote that a point $\mathbf{p}$ is inside the polygon $\boldsymbol{\rho}^e$ that defines the shape of e. Notice that $\boldsymbol{\rho}^e$ is transformed to a new representation defined by $\mathbf{b}^e$ and $\boldsymbol{\xi}^e$, which is more convenient for ANNs.

Accordingly, Equation (5.11) can be re-written in terms of the joint probability of the parts of each layout region as:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{*}{=} \prod_{e \in h} \mathrm{P}(\mathbf{b}^e, c^e, \boldsymbol{\xi}^e \mid X = \mathbf{x}) \tag{5.12}$$

Furthermore, assuming each part of e is independent of the other parts, the Equation (5.12) can be further simplified to:

$$\mathrm{P}(h \mid X = \mathbf{x}) \overset{*}{=} \prod_{e \in h} \mathrm{P}(\mathbf{b}^e \mid X = \mathbf{x})\, \mathrm{P}(c^e \mid X = \mathbf{x})\, \mathrm{P}(\boldsymbol{\xi}^e \mid X = \mathbf{x}) \tag{5.13}$$

Consequently, $\mathrm{P}(h \mid X = \mathbf{x})$ can be modeled independently for each part of the layout region e as a multi-task problem.

5.1.2.1 Training Process

As we were able to formulate the *Direct* approach to the Region Segmentation problem in the same terms as the Baseline Detection problem, the training procedure is interchangeable. Furthermore, Equation (4.15) and Equation (4.16) can be re-written using Region Segmentation variables and updated ANN output functions ($\boldsymbol{\Phi}_{\mathbf{b}^e} : \mathbb{R}^d \to \mathbb{R}^4$, $\boldsymbol{\Phi}_{c^e} : \mathbb{R}^d \to [0,1]^{|\mathcal{Y}|+1}$, $\boldsymbol{\Phi}_{\boldsymbol{\xi}^e} : \mathbb{R}^d \to [0,1]^{d'_{\boldsymbol{\xi}^e}}$) as follows:

$$\begin{aligned} J(\boldsymbol{\theta}) &= \lambda_{\mathbf{b}} J_{\mathbf{b}} + \lambda_c J_c + \lambda_{\mathbf{m}} J_{\mathbf{m}} \\ &= \frac{1}{N} \sum_{n=1}^{N} \left(\lambda_{\mathbf{b}} \frac{1}{|h_n|} \sum_{e \in h_n} \| \mathbf{b}^e - \boldsymbol{\Phi}_{\mathbf{b}^e}(\mathbf{x}_n) \| \right. \\ &\quad + \lambda_c \frac{-1}{|h_n|} \sum_{e \in h_n} \frac{1}{|\mathcal{Y}'|} \mathbf{c}'^e \log(\boldsymbol{\Phi}_{c^e}(\mathbf{x}_n)) \\ &\quad \left. + \lambda_{\boldsymbol{\xi}} \frac{-1}{|h_n|} \sum_{e \in h_n} \frac{1}{d'_{\boldsymbol{\xi}^e}} \sum_{k \in \boldsymbol{\xi}^e} k \log(\Phi_{\boldsymbol{\xi}^e}(\mathbf{x})) + (1-k) \log(1 - \Phi_{\boldsymbol{\xi}^e}(\mathbf{x})) \right) \end{aligned} \tag{5.14}$$

$$J(\boldsymbol{\theta}) = \lambda_{\mathbf{b}} J_{\mathbf{b}} + \lambda_c J_c + \lambda_{\boldsymbol{\xi}} J_{\boldsymbol{\xi}} + \lambda_{\mathrm{RPN}} J_{\mathrm{RPN}} \tag{5.15}$$

where $\mathcal{Y}' = \{\mathcal{Y}, 0\}$ is the set of all classes plus the background class, and $\mathbf{c}'^e \in \{0,1\}^{|\mathcal{Y}|+1}$ is the one-hot representation of the target value c^e.

5.1.2.2 Inference

Like in Section 4.1.2.2, let $\mathcal{B} = \{\beta_1, \beta_2, \ldots, \beta_n\}$ be the set of RoIs generated by an RPN. We can directly obtain its bounding box $\overset{*}{\mathbf{b}}^{\beta}$ and its class $\overset{*}{c}^{\beta}$ from Equation (5.16)[1] and Equation (5.17) respectively for each of them.

$$\overset{*}{\mathbf{b}}^{\beta} = \boldsymbol{\Phi}_{\mathbf{b}^{\beta}}(\mathbf{x}) \tag{5.16}$$

$$\overset{*}{c}^{\beta} = \underset{i \in \{c \in \mathcal{Y}, 0\}}{\arg\max}\, \Phi_{c^{\beta}, i}(\mathbf{x}) \tag{5.17}$$

On the other hand, the polygon that defines the shape of each predicted layout element $\boldsymbol{\rho}^{\beta}$ can be obtained by the same ConnectedComponents procedure defined on the *Map-based* approach over the most probable element map $\overset{*}{\boldsymbol{\xi}}^{\beta}$.

Finally, the most probable document layout can be retrieved from the most probable set of RoIs using Equation (5.18) as the subset of $\mathcal{B}$ which has been classified to belong to any class $c \in \mathcal{Y}$ (i.e., not background).

$$\overset{*}{h} = \{\beta \in \mathcal{B} \mid \overset{*}{c}^{\beta} \neq 0\} \tag{5.18}$$

5.2 Integrated Approach

Having a system able to perform Baseline Detection and another that performs Region Segmentation is very useful as we can obtain valuable data that can be used to obtain accurate information from documents. However, from a practical point of view, having two different models implies twice the effort to obtain such valuable data (i.e., time to train the models, time to run inference, memory used, etc.). Consequently, it is of interest to merge both systems into a single system that reduces such effort as much as possible.

Indeed, as mentioned at the beginning of this chapter, it is straightforward to extend the proposed formulation to handle both Baseline Detection and Region Segmentation problems in an integrated way.

By an integrated way, we mean that both problems can be modeled by the same statistical model (a.k.a an ANN) at the same time by sharing most of the model parameters.

Under those circumstances, one can extend the proposed methods under the Multitask Learning (MTL) framework [Car93], where each problem is considered a *task* that may help the other as an inductive bias. Hence, the integrated model will be trained to learn both tasks simultaneously (using a composed loss function).

[1] Notice that Equation (5.16) and Equation (4.17) are both the same equation, although we reproduce it here for convenience.

5.2.1 Map-based Approach

In both Baseline Detection and Region Segmentation problems, the input to the statistical models (ANNs) is the same, an image $\mathbf{x}$, while the main difference is the prediction layer. Then, it is very likely that the feature maps learned by the models on the first layers would be very similar.

Following that idea, it is straightforward to merge both problems under the MTL framework by keeping the first layers of the ANN as a set of shared layers, while the prediction layers are defined as a separate branch for each task. This approach is depicted in Figure 5.2, where the *feature extraction* layers are shared across tasks.

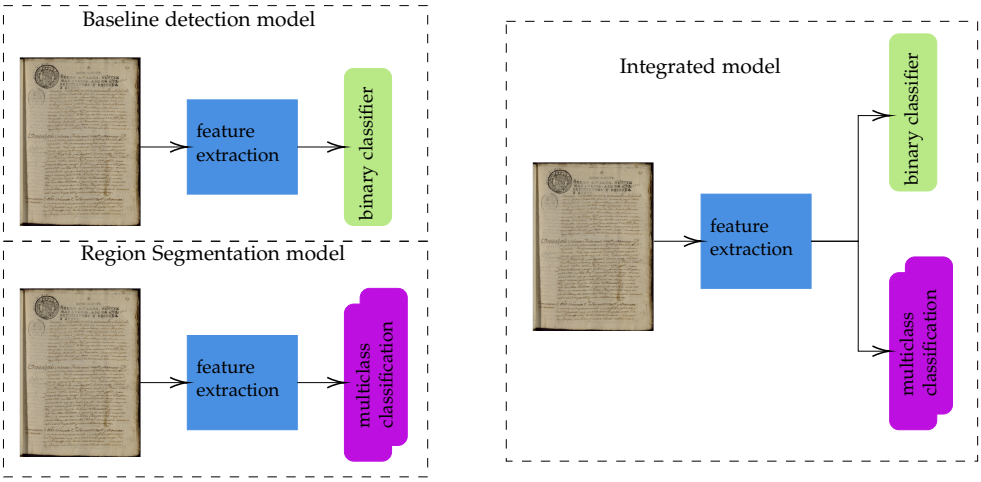


Figure 5.2: Integrated composition of the Map based approach model.

Of course, in order to train the integrated model, we use a composed loss function that is defined as the sum of the loss function of each task (Equation (4.7) and Equation (5.7)):

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\frac{-1}{d'}\left(\sum_{\mathbf{p}\in\mathbf{x}_n} m_{n,\mathbf{p}}\log(\Phi^1_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta})) + (1-m_{n,\mathbf{p}})\log(1-\Phi^1_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta})) + \sum_{\mathbf{p}\in\mathbf{x}_n}\boldsymbol{\zeta}'_{n,\mathbf{p}}\log(\boldsymbol{\Phi}^2_{\mathbf{p}}(\mathbf{x}_n,\boldsymbol{\theta}))\right) \tag{5.19}$$

where $\Phi^t_{\mathbf{p}}$ is the output of the network at point $\mathbf{p}$ for task t, being $t = 1$ for Baseline Detection and $t = 2$ for Region Segmentation; $\Phi^1_{\mathbf{p}} : \mathbb{R}^d \to [0,1]^{d'}$ and $\boldsymbol{\Phi}^2_{\mathbf{p}} : \mathbb{R}^d \to [0,1]^{d'\times|\mathcal{Y}|+1}$.

Finally, once the integrated model is trained, the inference is made for each task separately by the same formulation defined for each task in Section 4.1.1.2 and Section 5.1.1.2, respectively.

5.2.2 Direct Approach

In the *Direct* approach, an integrated model is even more straight to obtain as it is already defined as an MTL problem (i.e., each part of e is considered as a different task). Indeed, the baselines can be seen as a layout region with a different class (for instance, let us call it textline). Hence, a new set of classes $\mathcal{Y}' = \{\mathcal{Y}, \text{textline}\}$ can be used to train a single ANN to predict RoIs belonging those classes, with a map defined as follows:

$$\xi^e = \begin{cases} \mathbf{m}^e = 1 \iff d(\mathbf{p}, e) < \delta, \mathbf{p} \sqsubseteq \mathbf{b}^e, c^e = \text{textline} \\ 1 \iff \mathbf{p} \sqsubseteq \boldsymbol{\rho}^e, c^e \neq \text{textline} \end{cases} \tag{5.20}$$

It is important to notice that, since we did not change the structure of the statistical model but extend the types of regions of interest to be predicted, the loss function defined in Equation (5.14), and therefore in Equation (5.15), holds for the integrated approach as well.

Finally, during inference time, those RoIs predicted to belong to the class textline are processed using the baseline detection procedure defined in Section 4.1.2.2, while the others are still processed as a normal layout regions as defined in Section 5.1.2.2.

In Section 7.5 we experimentally evaluate this integrated approach on handwritten documents. Also, we compare this approach with to respect to the non-integrated approaches defined in Section 4.1 and Section 5.1.

6 *Reading Order Determination*

Although modern ATR systems can obtain very good results at recognizing the text that is written in a paper, most modern systems are designed to process only a text line at a time. Hence, the data obtained from each layout element needs to be arranged in order to obtain the full meaning of that data. The most common order in which those layout elements should be ordered is called the reading order.

For instance, let us analyze the case of Figure 6.1, in which five layout elements are depicted (baselines). At first glance, one might think that the correct way to read the document is first A, then C, then B, then E and finally D, as in most Latin scripts the tendency is to read from top-to-bottom-left-to-right (TBLR). However, a detailed analysis of the text will lead us to read first A, then C, then E, then B and finally D, since the elements $\{C, E\}$ and $\{B, D\}$ are related.

This is a common example of a document where following simple geometrical rules (like TBLR) does not work, moreover, depending on how the distance between elements is defined it could lead to different orders (e.g., A, B, C, D, E if we use the center of each baseline as reference). In many handwritten text documents, the correct reading order also depends on the type of region each text line belongs to (paragraph, page-number, marginalia, etc.) and maybe other criteria which are

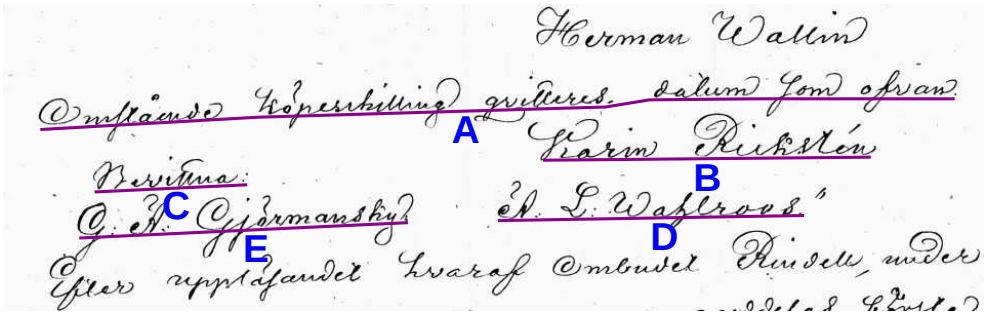


Figure 6.1: Example of text lines where a naive top-to-bottom-left-to-right order would result in reading A before B before C before D before E, while the correct reading order is A before C before E before B before D.

heavily dependent on the types of documents considered. Clearly, in such cases, the naive approach fails.

In our work, we adopt a novel viewpoint for this problem: to automatically sort layout elements of handwritten documents into reading order, first, an order-relation operator is learned from examples, then this operator is used to decode the unknown reading order of new documents.

Remember that a layout element e will generally have a different representation depending on the type of element. For instance, if the layout element is a baseline, it will be represented by a PLC, or if it is a layout region it should rather be represented by a polygon[1].

Formally, let $S = \{e_1, e_2, \ldots, e_n\}$ be a set of n layout elements of a page image. An order in S can be defined by a *permutation*, usually represented by a set $\mathbf{z}$ of n pairs $\{(e, \nu) : e \in S, \nu \in \mathbb{N}\}$, where ν, called "index", also satisfies $\nu_i \neq \nu_j, 1 \leq i, j \leq n \; \forall j \neq i$.

For instance, for the document in Figure 6.1, the permutation $\mathbf{z}' = \{(A,1),(B,2),(C,3),(D,4),(E,5)\}$ represents a naive TBLR order in S, while the permutation representing the correct reading order is $\hat{\mathbf{z}} = \{(A,1),(B,4),(C,2),(D,5),(E,3)\}$.

Our problem can be now stated as follows: *Given a set of layout elements S, obtain a permutation $\mathbf{z}$ of S that renders its elements in reading order*.

In general, the best reading order $\mathbf{z}^\star$ is a solution to the following optimization problem:

$$\mathbf{z}^\star = \arg\max_{\mathbf{z} \in Z} \mathrm{P}(\mathbf{z}) \tag{6.1}$$

where Z is the set of all possible permutations[2] of S and $\mathrm{P}(\mathbf{z})$ is the probability that $\mathbf{z}$ renders S in the correct reading order.

Regularly, it is useful to describe $\mathbf{z}$ in matrix form. To that end, $\mathbf{z}$ can be described in terms of the absolute position of each element in the permutation, or in terms of the relative position of each element with respect to the others. We found the latter more suitable for further development.

Let $e_i, e_j \in S$ be two layout elements. An order in S can alternatively be specified by means of a binary order relation $\prec$ on $S \times S$:

$$e_i \prec e_j \text{ means } e_i \text{ "is placed before" } e_j \tag{6.2}$$

which is assumed to fulfill all the properties of a strict total order [DP90].

For instance, for the permutation $\hat{\mathbf{z}} = \{(A,1),(B,4),(C,2),(D,5),(E,3)\}$, we can state: $A \prec C$, $C \prec E$, $E \prec B$, $B \prec D$ and also, $A \prec E$, $A \prec B$, $A \prec D$, $C \prec B$, $C \prec D$, $E \prec D$.

[1] Several geometrical features can be extracted from those representations in order to feed any statistical model. We provide mode details about those features in Section 7.2.2.3.

[2] Since $|Z| = n!$ the complexity of the problem to rapidly increases with the size of S.

Therefore, this binary order relation can be represented in matrix form as shown in these examples:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad \cdots \quad \begin{array}{c} \\ A \\ C \\ E \\ B \\ D \end{array}\!\!\begin{array}{c} \begin{array}{ccccc} A & C & E & B & D \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \tag{6.3}$$

Note that these matrices represent the very same binary order relation $\prec$ (the one associated with the correct order defined by $\hat{\mathbf{z}}$). However, the left one is organized with rows and columns ordered according to a naive order (given by $\mathbf{z}'$), while in the right one rows and columns follow the canonical sequential rendering of $\mathbf{z}$, which is always of the form [3] $\mathbf{z} = \{(e_1, 1), \ldots, (e_n, n)\}$. In general, we call this matrix $R^{\mathbf{z}} = [r^{\mathbf{z}}_{i,j}]^{n \times n}$, where $r^{\mathbf{z}}_{i,j}$ is defined in terms of the indexes of $\mathbf{z}$ as:

$$r^{\mathbf{z}}_{i,j} = \begin{cases} 1 & \text{if } \nu_i < \nu_j \,,\, \nu_i, \nu_j \in \mathbf{z} \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

Notice that the indexes defining a permutation $\mathbf{z}$ can be straightforwardly computed from the corresponding matrix by just counting the number of zeros of each row, and its canonical form can be obtained by just sorting $\mathbf{z}$ in ascending numerical order according to the indexes ν_i, $1 \leq i \leq n$. We will capitalize on this observation later to propose a very fast decoding algorithm to determine the reading order using the results of learning $\prec$ from training examples.

We can re-write Equation (6.1) in terms of its matrix form $R^{\mathbf{z}}$ as:

$$\mathbf{z}^\star = \arg\max_{\mathbf{z} \in Z} \mathrm{P}(R^{\mathbf{z}}) \tag{6.5}$$

where $\mathrm{P}(R^{\mathbf{z}})$ can be expressed as the joint probability of all its elements, and applying the chain rule of probability:

$$\mathrm{P}(R^{\mathbf{z}}) = \mathrm{P}(r^{\mathbf{z}}_{1,1})\, \mathrm{P}(r^{\mathbf{z}}_{1,2} \mid r^{\mathbf{z}}_{1,1}) \cdots \mathrm{P}(r^{\mathbf{z}}_{n,n} \mid r^{\mathbf{z}}_{1,1}, \ldots, r^{\mathbf{z}}_{n,n-1}) \tag{6.6}$$

Now, we assume that the relationship between a pair of elements in S (i.e., $r^{\mathbf{z}}_{i,j}$) is independent of the relationship between any other pair of elements:

$$\mathrm{P}(R^{\mathbf{z}}) \approx \prod_{i=1}^{n} \prod_{j=1}^{n} \mathrm{P}(r^{\mathbf{z}}_{i,j}) \tag{6.7}$$

[3] Note that the sub-indexes of the elements e_i in this expression do *not* express any order, they are just an identifier of the element.

Remember that the elements of $R^{\mathbf{z}}$ satisfy $r_{i,j} = 1 - r_{j,i}$, $1 \leq i, j \leq n$, $i \neq j$. Therefore, $R^{\mathbf{z}}$ is completely defined by its upper triangular part:

$$\mathrm{P}(R^{\mathbf{z}}) \approx \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \mathrm{P}(r_{i,j}^{\mathbf{z}})^2 \tag{6.8}$$

Finally, form Equation (6.1) and Equation (6.8):

$$\mathbf{z}^{\star} \approx \arg\max_{\mathbf{z} \in Z} \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \mathrm{P}(r_{i,j}^{\mathbf{z}}) \tag{6.9}$$

where the square operator over $\mathrm{P}(r_{i,j}^{\mathbf{z}})$ is dropped as it does not affect the maximization process.

Notice that with respect to directly using Equation (6.7), using Equation (6.9) reduces the number of required multiplications from n^2 to $(n^2 - n)/2$.

Now, we need to estimate $\mathrm{P}(r_{i,j}^{\mathbf{z}})$ and efficiently decode $\mathbf{z}^{\star}$ from Equation (6.9). To that end, in Section 6.1 a way to learn $\mathrm{P}(r_{i,j}^{\mathbf{z}})$ from training samples is defined, and in Section 6.2 two efficient decoding methods are presented.

6.1 Learning the Pairwise Binary Order Relation

The binary order relation $\prec$ defined in Equation (6.2) can be explicitly re-written as a function $\mathcal{Q} : S \times S \rightarrow \{0, 1\}$. Therefore, learning to sort layout elements into a correct reading order amounts to learn $\mathcal{Q}$ from training examples of its input pair and binary output. To this end, we need training examples of page images with correctly sorted layout elements, which are generally available as a byproduct of annotating text images with their correct transcripts. Let $\mathbf{z} = \{(e_1, \nu_1), \ldots, (e_n, \nu_n)\}$ be the permutation corresponding to a correctly sorted set of layout elements of a page image. From this permutation, we construct the training samples of $\mathcal{Q}$ for all the possible pairs e_i, e_j as:

$$([e_i, e_j], y), \; y = \begin{cases} 1 & \text{if } \nu_i < \nu_j \,,\, 1 \leq i, j \leq n \,,\, i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{6.10}$$

For example, given the document in Figure 6.1 with ground-truth reading order $\hat{\mathbf{z}} = \{(A, 1), (B, 4), (C, 2), (D, 5), (E, 3)\}$, the following set of training samples will be generated:

$$([A, C], 1), \; ([A, E], 1), \; ([A, B], 1), \; ([A, D], 1),$$

$$
\begin{aligned}
([C,E],1),\ ([C,B],1),\ ([C,D],1),\\
([E,B],1),\ ([E,D],1),\\
([B,D],1),\\
([C,A],0),\ ([E,A],0),\ ([B,A],0),\ ([D,A],0),\\
([E,C],0),\ ([B,C],0),\ ([D,C],0),\\
([B,E],0),\ ([D,E],0),\\
([D,B],0)
\end{aligned}
\tag{6.11}
$$

This process is applied to the correctly sorted layout elements of all available training images, resulting in a training set that will be referred to as $\mathcal{D}$.

From $\mathcal{D}$, a model is trained to estimate the conditional distribution $\mathrm{P}(Y = y \mid e_i, e_j)$, where Y is a Boolean random variable such that $Y = 1$ *iff* $e_i \prec e_j$, and e_i, e_j is the value of a random variable corresponding to an ordered pair of layout elements. Therefore, note that e_i and e_j are not interchangeable in $\mathrm{P}(y \mid e_i, e_j)$; that is e_i, e_j should not be seen as a conventional conjunction of two individual conditions but as a single condition by itself.

It may happen that this estimated distribution lacks some properties which would be desired for a proper order-relation probabilistic model. For instance, it may come about that $P(Y = 1 \mid e_i, e_j) > 0.5$ (from which we may infer that $e_i \prec e_j$), and also $P(Y = 0 \mid e_j, e_i) < 0.5$ (which suggests just the opposite, $e_j \prec e_i$). This problem is most likely caused by the fact that $P(y \mid e_i, e_j)$ is only conditioned by two specific layout elements, ignoring the rest of the layout elements of the image. Ideally, for any $e_i, e_j \in S$, $P(Y = 1 \mid e_1, \ldots, e_i, e_j, \ldots, e_n)$ should be identical to $P(Y = 0 \mid e_1, \ldots, e_j, e_i, \ldots, e_n)$. In our experiments, we have rather seldom found that $P(Y = 1 \mid e_i, e_j)$ and $P(Y = 0 \mid e_j, e_i)$ are different. However, better overall performance can be achieved if we enforce strict equality by heuristically re-estimating the pairwise order probability as the average of $P(Y = 1 \mid e_i, e_j)$ and $P(Y = 0 \mid e_j, e_i)$:

$$
\tilde{P}(y \mid e_i, e_j) = \frac{P(y \mid e_i, e_j) + 1 - P(y \mid e_j, e_i)}{2},\ y \in \{0, 1\},\ e_i \neq e_j \tag{6.12}
$$

And, obviously, $\tilde{P}(Y = 1 \mid e_i, e_i) = 0$, $\tilde{P}(Y = 0 \mid e_i, e_i) = 1$, $\forall e_i \in S$. Correspondingly, since $\tilde{P}(Y = 1 \mid e_i, e_j) + \tilde{P}(Y = 0 \mid e_i, e_j) = 1\ \forall e_i, e_j \in S$, $\tilde{P}(Y = y \mid e_i, e_j)$ can still be properly interpreted probabilistically.

The values of $\tilde{P}(Y = 1 \mid e, e')$ can be arranged in matrix form, exactly as in the examples of Equation (6.3). In fact, those matrices can be seen as the values of $P(Y = 1 \mid e_i, e_j)$, where probabilities only have the extreme values 0 or 1. Hence, $\mathrm{P}(r_{i,j})$ can be estimated directly from Equation (6.12):

$$
\mathrm{P}(r_{i,j}) = \tilde{P}(Y = r_{i,j} \mid e_i, e_j) \tag{6.13}
$$

Moreover, Equation (6.9) can be re-written in terms of Equation (6.13) as:

$$\mathbf{z}^{\star} \approx \arg\max_{\mathbf{z} \in Z} \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \tilde{P}(Y = r_{i,j} \mid e_i, e_j) \tag{6.14}$$

Provided that, $\tilde{P}(Y = r_{i,j} \mid e_i, e_j)$ can be estimated by means of a binary classifier. For instance, in [QV21] we estimate it using and MLP and a Random Forest Classifier (RFC). Moreover, in Section 7.6 we experimentally estimate this probability for few datasets.

A few illustrative examples of possible pairwise order relation probability matrices for the five lines example of Figure 6.1 are given in Appendix A.

6.2 Decoding a Best Reading Order

Once the pairwise binary order-relation is estimated by some model, we discuss how to obtain the reading order of the set of layout elements of any new, unseen, text image.

The most straightforward decoding method is by brute force, that consists in checking all the $n!$ permutations [4] of S, which makes it intractable for most practical scenarios. Hence, two decoding methods are presented below to cope with the huge complexity of this problem.

6.2.1 Greedy Decoding

Since global optimization is too computationally expensive for most real cases, we can resort to local optimization. A good approximation can be obtained by selecting the best candidate as the most probable layout element at each position $t, 1 \leq t \leq n$, using Algorithm 4.

Notice that this algorithm can generate a sub-optimal solution to the main problem, as we demand the solution only to be locally optimal on each position of the reading order.

By construction, Algorithm 4 ensures that the resulting permutation, $\tilde{\mathbf{z}}^{\star}$, is *proper*. However, this greedy method is sub-optimal and therefore the probability of $\tilde{\mathbf{z}}^{\star}$ is just a lower bound of the probability of the optimal permutation, that is:

$$P(\tilde{\mathbf{z}}^{\star}) \leq \mathrm{P}(\mathbf{z}^{\star}) \tag{6.15}$$

Nevertheless, according to our observations and results reported in Section 7.6, the bound is fairly tight. Therefore, if $\tilde{P}(Y = y \mid e, e')$ is well estimated, good approximations to the global optimum are generally retrieved.

[4] A common page of a handwritten text document could range from few elements (e.g., $n = 20$, $n! = 2.4 \times 10^{18}$) to hundreds of elements (e.g., $n = 200$, $n! = 7.8 \times 10^{374}$).

Algorithm 4 Greedy approximative algorithm to decode the most probable reading order.

Require: a set of layout elements ($S = \{e_1, \ldots, e_n\}$), its pairwise probability ($\tilde{P}(Y = 1 \mid e, e'), \forall e, e' \in S$)

Output: locally optimum reading order ($\tilde{\mathbf{z}}^\star$)

$t = 1$
$\tilde{\mathbf{z}}^\star \leftarrow \{\emptyset\}$
while $t \leq n$ **do**
  $b \leftarrow 0$
  for $e \in S$ **do**
    $c \leftarrow \prod_{e' \in S, e \neq e'} \tilde{P}(Y = 1 \mid e, e')$ ▷Probability of the element e to be placed before any other element $e' \in S$.
    if $c > b$ **then**
      $s \leftarrow e$ ▷keep the most probable element.
      $b \leftarrow c$
    end if
  end for
  $\tilde{\mathbf{z}}^\star \leftarrow \tilde{\mathbf{z}}^\star \cup \{(s, t)\}$ ▷add the most probable element to the permutation $\tilde{\mathbf{z}}^\star$ at position t.
  $S \leftarrow S \setminus \{s\}$ ▷remove the most probable element from the search space.
  $t{+}{+}$
end while
return $\tilde{\mathbf{z}}^\star$

In Appendix A we provide numerical examples of results obtained with this method for the five lines ecample of Figure 6.1.

6.2.2 First Decide then Decode (FDTD) Decoding

Greedy decoding is useful to overcome the computational complexity of searching for the global optimal solution. However, the decoding process can be further simplified and the accuracy of results improved if we follow the observation mentioned at the beginning of this chapter in relation to the example of Figure 6.1 and Equation (6.3).

Following Equation (6.7), an equation similar to Equation (6.14) can be written without explicit permutation notation as follows:

$$R^\star \approx \arg\max_R \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} \tilde{P}(Y = r_{i,j} \mid e_i, e_j) \tag{6.16}$$

where $R = [r_{i,j} \in \{0, 1\}]^{n \times n}$, with $r_{i,i} = 0, 1 \leq i \leq n$.

Notice that without any restriction of the relative values of the elements of R, a straightforward solution to the optimization problem (Equation (6.16)) is achieved for a matrix such that each individual factor of the product is maximum. This allows

us to solve Equation (6.16) by simply setting for $1 \leq i, j \leq n$:

$$r^{\star}_{i,j} = \underset{y=\{0,1\}}{\arg\max}\, \tilde{P}(Y = y \mid e_i, e_j) \equiv \begin{cases} 1 & \text{if} \quad \tilde{P}(Y = 1 \mid e_i, e_j) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{6.17}$$

That is, instead of decoding the best solution directly from Equation (6.16), we *first decide* which element in S is to be placed before the other (Equation (6.17)). Then, as mentioned at the beginning of this chapter, given $R^{\star}$, the indexes $(v^{\star}_i)$ of the corresponding permutation can be obtained by just counting the number of zeros in the i-th row of $R^{\star}$; that is:

$$v^{\star}_i = \sum_{j=1}^{n} (1 - r^{\star}_{i,j}),\ 1 \leq i \leq n \tag{6.18}$$

It is important to realize, however, that the permutation computed in this way from a matrix $R^{\star}$, although optimal for Equation (6.16), may be *improper*.

Evidently, since $\tilde{P}(y \mid e, e') = 1 - \tilde{P}(y \mid e', e)$, the elements of $R^{\star}$ satisfy $r^{\star}_{i,j} = 1 - r^{\star}_{j,i}, 1 \leq i, j \leq n, i \neq j$, which is one of the conditions required for $R^{\star}$ to represent a proper permutation. Yet, this matrix may still fail to represent a proper permutation for other more subtle reasons. For instance, since $\tilde{P}(Y = y \mid e_i, e_j)$ does not take into account the complete context of S, the following cases can arise:

- $\tilde{P}(Y = 1 \mid e_i, e_j) > 0.5$,
- $\tilde{P}(Y = 1 \mid e_j, e_k) > 0.5$,
- $\tilde{P}(Y = 1 \mid e_k, e_i) > 0.5$.

The first two are normal cases which imply that e_i should be placed before e_j and e_j before e_k. However, the last one, implies that e_k should be placed before e_i, which contradicts the transitivity property of a total order. As a result, Equation (6.17) leads to ties in the number of zeros per row in $R^{\star}$, and Equation (6.18) yields repeated values of $v^{\star}$.

Clearly, if $R^{\star}$ does not have ties, the corresponding permutation is an optimal solution to Equation (6.9). Otherwise, note that the optimization problem 6.16 is equivalent to a relaxed version of problem 6.9 where one of the restrictions has been dropped. Therefore, the probability of Equation (6.8) computed for $R^{\star}$ is an upper bound of the maximum probability associated with the optimal $\mathbf{z}^{\star}$ of Equation (6.9). That is:

$$P(R^{\star}) \ \geq \ \mathrm{P}(R^{\mathbf{z}^{\star}}) \tag{6.19}$$

Moreover, according to our observations and results reported in Section 7.6, the bound is fairly tight. This suggests that, even in the cases where $R^{\star}$ does not

correspond to a proper permutation, we can still obtain a close to optimal (proper) permutation by adequately breaking the ties in the number of zeros per row in $R^\star$. While several heuristics can be used for this purpose, in Section 7.6 we just solve the few ties observed arbitrarily, in order to keep the experimentation as comprehensible as possible.

This approach is illustrated using numerical examples in Appendix A for the five lines example of Figure 6.1.

6.3 Hierarchical Approach

Although the proposed model has been described at page level, it can be used in any hierarchical approach, where Equation (6.12) and any suitable decoder will be applied in each hierarchical level independently.

First, if there is no tacit interest in sorting all the layout elements of a document, but, for instance, only the text lines inside the layout regions. This local sorting will reduce the complexity of the problem as the number of elements inside a layout region is expected to be lower than the number of elements at page level. Secondly, as the problem complexity increases exponentially with the number of input pairs, a hierarchical processing of the data will reduce the complexity as we solve several small problems instead of a big one.

Nevertheless, beware that any error in a higher level will place all its sub-elements in the wrong position, increasing the risk of obtaining an incorrect reading order. However, this can also make the correction process simpler, since a single fix in a given level may lead to correctly sort all the corresponding sub-elements in the lower level.

In Section 7.6, experiments are provided in order to compare and analyze the present approach at different hierarchical levels.

7 Experiments

In this chapter we will experimentally validate our probabilistic framework, algorithms and models, described trough the previous chapters. In particular, some main questions that we aim to answer trough the different proposed experiments are:

- What is the performance of the *Map-based* approach and the *Direct* approach to address the Baseline Detection problem on handwritten documents? We examine this topic in Section 7.3.1 using textual and musical handwritten documents.
- Whats is the performance of the *Map-based* approach and the *Direct* approach to address the Region Segmentation problem on handwritten documents? This topic is developed in Section 7.4.1, also for textual and musical handwritten documents.
- How does the proposed systems perform under restricted amount of training samples? This topic is examined in Section 7.3.2 and Section 7.4.2, when addressing the Baseline Detection and Region Segmentation problems, respectively.
- It is possible to handle both Baseline Detection and Region Segmentation problems in an integrated way? What is the effect of the integrated setting in the performance of the proposed models? Both questions will be addressed in Section 7.5.
- How does the models proposed for Baseline Detection and Region Segmentation problems perform on mixed datasets? This in an interesting topic for production scenarios that we examine in Section 7.5.1.
- How the size of the input sample influence the behavior of the proposed decoding algorithms for the Reading Order Determination problem? We consider this issue from the computational complexity of the algorithms in Section 7.6.1.

- What is the performance of the proposed methods to address the Reading Order Determination problem? This topic is examined in Section 7.6.2 using textual handwritten documents.
- Is a hierarchical approach of the Reading Order Determination problem feasible? This question is addressed in Section 7.6.3, where we also compare it to the non-hierarchical approach.
- How our proposed systems compares with SOTA works in the field? We analyze this question for the Baseline Detection and Region Segmentation problems in Section 7.3.3 and Section 7.4.3, respectively.

7.1 Experimental Setup

7.1.1 Databases

In order to compare our work to other approaches, we carried our experiments on several publicly available databases whenever it is possible. In addition, when there is no public database available with the necessary characteristics to evaluate the proposed method, we create a new one (or enhance a public one) and make it publicly available.

In this thesis, we do not limit ourselves to the very common textual handwritten documents, but we also focus on musical handwritten documents. With this intention, we evaluate our methods in two main databases, namely *Oficio de Hipotecas de Girona* (OHG), and Vorau Abbey library Cod. 253 (VORAU-253). Nonetheless, we will also report results on other databases for completeness in the comparison with previous SOTA works.

In the following paragraphs, we briefly introduce thse two databases. More details and figures about the content of the documents can be found in Appendix B, while we refer the reader to the corresponding publications for a detailed description about the data collection and ground-truth preparation.

***Oficio de Hipotecas de Girona* (OHG)** database was created by us due to the lack of a publicly available database of handwritten text documents annotated at both baseline and region level and with a coherent reading order.

The publicly available database [Qui+18b] is a portion of 596 pages from the collection, from batch *b001* to batch *b012*. OHG pages exhibit a relatively complex layout, composed of six relevant region types; namely: `pag`, `tip`, `par`, `pac`, `not`, `nop`, as described in detail in Appendix B.1.

The data is randomly divided into 298 images for training and 298 for test. An example is depicted in Figure 7.1 and the main characteristics are summarized in

Table B.2, from which we can extract that the number of lines per page is 39.9 [33, 66] on average, distributed over an average of 4.8 [1, 11] layout regions per page.

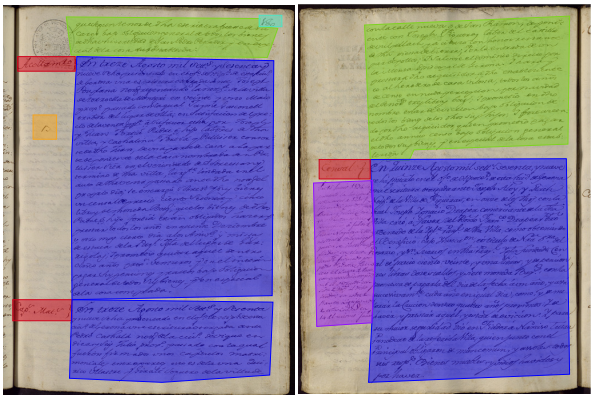


Figure 7.1: Examples of pages with different layouts, belonging to the OHG database. Cyan: `pag`, red: `tip`, green: `pac`, blue: `par`, violet: `not`, orange:`nop`.

Vorau Abbey library Cod. 253 (VORAU-253) database is a beautiful illuminated musical manuscript originally labeled for the Handwritten Music Recognition (HMR) task. Hence, only the transcript of some staff regions were annotated. Provided that, we augmented the available ground-truth by labeling the layout of 228 images into three different layout regions; namely: `lyrics`, `staff` and `drop-capital`, along with the corresponding baselines of the text found in the `lyrics` regions. A detailed description of the dataset is provided in Appendix B.2.

The dataset was randomly divided into 128 images for training and 100 for test. An example is depicted in Figure 7.2 and the main characteristics are summarized in Table B.3.

7.1.2 Evaluation Protocol

To the best of our knowledge, there is no common evaluation measure capable of jointly evaluating the results obtained in all the tasks addressed in this thesis[1]. Therefore, in the next subsections, we present a set of metrics for each task.

[1] Due to the fact that the most common application of DLA is to feed some ATR system, it is possible to indirectly evaluate the results obtained trough ATR results by computing the CER, WER [MV93], BLEU [Pap+02] or any other metric at transcript level. Nonetheless, an indirect approach will lead to many difficult challenges like the alignment between ground-truth and the hypotheses, the error propagation trough DLA/ATR systems, etc.

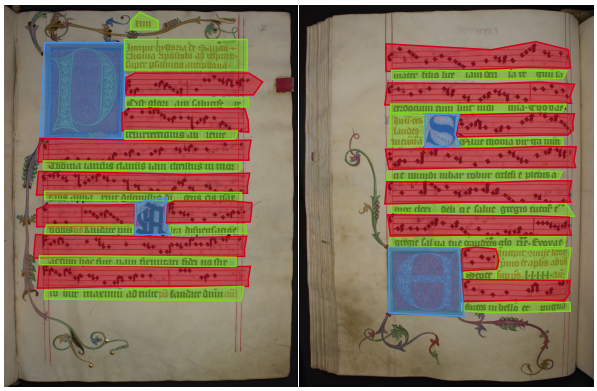


Figure 7.2: Examples of pages with different layouts, belonging to the VORAU-253 database. Blue: `drop-capital`, red: `staff`, green: `lyrics`.

When appropriate, we provide confidence intervals (CI) of the studied statistics (e.g., precision, recall, Jaccard index, etc.). There are many ways to computing CI, but they require making assumptions about the statistic's distribution. However, those assumptions are not trivial to define, which can lead to possibly ill-founded approximations, and may not hold in may real scenarios. Provided that, we compute the CI using non-parametric bootstrapping [Efr87; BN04], which has the advantage that do not make any assumption about the distribution.

7.1.2.1 Baseline Detection Evaluation

We report P-value (P), R-value (R) and its harmonic mean (F1) measures as defined specifically for this kind of problem in [Grü+17]. Henceforth, let $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_n\}$ be a set of given ground-truth baselines (represented by a PLC) and $\mathcal{H} = \{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ be a set of hypothesis. The calculation of R and P for the two sets $\mathcal{G}$ and $\mathcal{H}$ follows the following procedure.

- Normalize $\mathcal{G}$ and $\mathcal{H}$, so that two adjacent vertexes are in the 8-neighborhood of each other. From here we use $\mathcal{G}$ and $\mathcal{H}$ as their normalized versions.
- Define a tolerance value $t_{\mathbf{g}}$ for each $\mathbf{g} \in \mathcal{G}$, such that baselines in the hypothesis which are slightly different to the ground-truth are not penalized.

$$t_{\mathbf{g}} = 0.25 \min(d_{\mathbf{g}}, \bar{d}_{\mathcal{G}}) \tag{7.1}$$

  where $d_{\mathbf{g}}$ is the minimum Euclidean distance from the vertexes of $\mathbf{g}$ to the vertexes of any other baseline in $\mathcal{G}$, and $\bar{d}_{\mathcal{G}}$ is the mean of all $d_{\mathbf{g}}, \mathbf{g} \in \mathcal{G}$.

- Compute the coverage function C. This function counts the number of vertexes of a normalized PLC for which there is a vertex of another normalized PLC with a distance less than the given tolerance value.

- The R-value is dependent of $\mathcal{G}$ and $\mathcal{H}$ and calculated by:

$$R(\mathcal{G},\mathcal{H}) = \frac{\sum_{\mathbf{g}\in\mathcal{G}} C(\mathbf{g},\mathcal{H},t_{\mathbf{g}})}{|\mathcal{G}|} \tag{7.2}$$

  The R-value indicates the fraction of the ground-truth baselines that are detected in the hypothesis within the tolerance value.

- The P-value should penalize segmentation errors, therefore an alignment between ground-truth and the hypothesis is needed. In this case the baselines are aligned based on the minimum distance between elements. Let $\mathcal{M}(\mathcal{G},\mathcal{H}) \in \mathcal{G}\times\mathcal{H}$ to be the alignment between its elements, then P is calculated as follows:

$$P(\mathcal{G},\mathcal{H}) = \frac{\sum_{(\mathbf{g},\mathbf{e})\in\mathcal{M}(\mathcal{G},\mathcal{H})} C(\mathbf{e},\mathbf{g},t_{\mathbf{g}})}{|\mathcal{H}|} \tag{7.3}$$

- Finally, the harmonic mean (F1) can be computed as:

$$F1 = \frac{2PR}{R+P} \tag{7.4}$$

We refer the reader to [Grü+17] for a description about the implementation and further details. In this thesis, we use the implementation provided by the authors of the metric[2].

Given the normalization of $\mathcal{G}$ and $\mathcal{H}$ in the first step of the process, the values of P, R and F1 cannot be longer interpreted in the classical way as related to the fraction of relevant baselines. Instead, they are related to the normalized vertexes of each baseline. For instance, classically P-value is understood as the fraction of baselines in the hypothesis that are relevant given the ground-truth, so $P = \frac{35}{40}$ is read as that 35 out of 40 baselines in the hypothesis are correct. But, due to the normalization, it should be interpreted as the fraction of normalized vertexes (i.e., not the whole baseline) in the hypothesis that are relevant. Hence, for the same P-value, it should be read as that 35 normalized sections out of 40 in the hypothesis are correct, where those 40 elements belong to one or more baselines. The same analysis should be done for R-value and the harmonic mean.

Moreover, in all our experiments we allow t_g to be estimated automatically from the ground-truth data, as it is the *de facto* configuration used by the scientific

[2] https://github.com/Transkribus/TranskribusBaseLineEvaluationScheme

community [Die+17; Die+19]. However, it is important to notice that, as pointed out by [Bos20], the value of this parameter could influence the relationship between the obtained results and the effect of that layout in further ATR systems. Hence, we recommend caution when inferring information from this evaluation scheme.

7.1.2.2 Region Segmentation Evaluation

We report metrics from semantic segmentation and scene parsing evaluations as presented in [LSD15]:

- Pixel accuracy (pixel acc.): $\frac{\sum_i \eta_{ii}}{\sum_i \tau_i}$
- Mean accuracy (mean acc.): $\frac{1}{|\mathcal{Y}|} \sum_i \frac{\eta_{ii}}{\tau_i}$
- Mean Jaccard Index (m^{IoU}): $\frac{1}{|\mathcal{Y}|} \sum_i \frac{\eta_{ii}}{(\tau_i + \sum_j \eta_{ji} - \eta_{ii})}$
- Frequency weighted Jaccard Index ($f.w.^{IoU}$): $(\sum_c \tau_c)^{-1} \sum_i \frac{\tau_i \eta_{ii}}{(\tau_i + \sum_j \eta_{ji} - \eta_{ii})}$

where η_{ij} is the number of pixels of class i predicted to belong to class j, $|\mathcal{Y}|$ is the number of different classes, τ_i the number of pixels of class i, and $c \in \mathcal{Y}$. Notice that there is no alignment between the target layout regions and the hypotheses hence they are global page-level metrics.

These are easy to compute metrics that help us to get a quick idea of the performance of the system as a pixel level classifier. However, as our goal is to obtain the layout regions, not only the pixels that belong to those regions, they give us a biased result, as they do not take into account if two or more layout regions were merged or if a region is split. Still, we keep using these metrics as they are very helpful to understand the performance of the statistical models used in the *Map-based* approach.

On the other hand, in order to quantitatively evaluate the performance and robustness of the proposed methods, taking into account the alignment between each layout region, we resort on standard *COCO Object Detection* metrics [Lin+15].

COCO Object Detection metrics are based on the well-known *precision*, *recall*, *mean average precision* (mAP) and the *Jaccard Index* (IoU). Forthwith, mAP is defined as:

$$\text{mAP} = \int_0^1 P(r)dr \tag{7.5}$$

where r represents a *recall* value, $P(r)$ denotes the precision value at which the r value corresponds to, and the *precision* and *recall* values are computed using the

standard formulas:

$$precision = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7.6}$$

$$recall = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7.7}$$

where TP (True Positives), FN (False Negatives) and FP (False Positives or false alarms) values are computed using IoU as an alignment rule as follows:

- TP is the number of predicted elements ($e_j \in \mathcal{H}$) where $\text{IoU}(e_i, e_j) > th$ for some ground-truth element e_i and threshold th, and $c^{e_i} = c^{e_j}$.
- FP is the number of predicted elements where there is no $\text{IoU}(e_i, e_j) > th, \forall e_i \in \mathcal{G}$ (i.e., e_j cannot be aligned to any element in the ground-truth).
- FN is the number of ground-truth elements where the is no $\text{IoU}(e_i, e_j) > th, \forall e_j \in \mathcal{H}$ (i.e., e_i cannot be aligned to any predicted element).

Unlike m^{IoU}, IoU is computed for each pair of layout elements $(e_i \in \mathcal{G}, e_j \in \mathcal{H})$ independently, instead of globally:

$$\text{IoU}(e_i, e_j) = \frac{\boldsymbol{\xi}^{e_i} \cap \boldsymbol{\xi}^{e_j}}{\boldsymbol{\xi}^{e_i} \cup \boldsymbol{\xi}^{e_j}} \tag{7.8}$$

In general, three sets of thresholds are used to compute mAP, namely[3]: AP, AP_{50} and AP_{75}. AP refers to mAP computed by averaging over a set of thresholds, from $th = 0.50$ to $th = 0.95$ with a step size of 0.05; AP_{50} refers to mAP computed using a single threshold $th = 0.50$; similarly $th = 0.75$ is used in the case of AP_{75}.

Following the same analysis, the *mean average recall* (mAR) can be defined in terms of *precision*, *recall* and Equation (7.8). We refer the reader to [Lin+15] for details about those metrics and its implementation. In this thesis, we use the official implementation of the metric, provided by the organizers of the competition[4].

7.1.2.3 Reading Order Evaluation

In order to evaluate the ordering returned by some automatic approach, a basic metric such as the number of misplaced elements or precision-recall is not enough. That kind of metrics will give us an idea of how many elements are misplaced but does not take into account how far those elements are from the correct position or

[3] Since this metrics are averaged over all classes, this metrics should formally be called mAP, mAP_{50} and mAP_{75}. However, it is common to assume that the difference is clear from context and AP versions are preferred instead.

[4] https://github.com/cocodataset/cocoapi

the estimated effort needed to manually correct any error. To that end, we resort to metrics used in information retrieval, specifically we consider the normalized Spearman footrule distance and the Kendall tau rank distance. Each metric is defined at page level[5] as follows:

Normalized Spearman footrule distance [KV10] is defined as the cumulative sum of distances between pairs in two ordering indexes as:

$$\rho(\mathbf{t}, \boldsymbol{\nu}) = \frac{\sum_{i=1}^{n} \mid t_i - \nu_i \mid}{\lfloor \frac{1}{2} n^2 \rfloor} \tag{7.9}$$

where $\mathbf{t}, \boldsymbol{\nu}$ are the ground-truth and the hypothesis ordering indexes for S, $\lfloor \frac{1}{2} n^2 \rfloor$ is the maximum cumulative distance possible between all pairs t_i, ν_i, and $0 \leq \rho(\cdot) \leq 1$.

Normalized Spearman footrule distance gives us the insight of not only how many elements are misplaced but how far are those elements from the correct position.

Kendall Tau Rank Distance [Ken38] is a metric also called bubble-sort distance, since it is equivalent to the number of swaps the bubble-sort algorithm would take to transform the order defined by $\boldsymbol{\nu}$ into the reference order defined by $\mathbf{t}$.

Formally, the absolute value of this metric is defined as the number of discordant pairs between $\mathbf{t}$ and $\boldsymbol{\nu}$:

$$K(\mathbf{t}, \boldsymbol{\nu}) = |\{(i,j) : i<j \wedge ((t_i<t_j \wedge \nu_i>\nu_j) \vee (t_i>t_j \wedge \nu_i<\nu_j))\}| \tag{7.10}$$

Note that $K(\mathbf{t}, \boldsymbol{\nu})$ help us to estimate the effort needed to fix any errors in the hypothesis. It is an upper bound to the number of edit operations a human would need to perform in order to obtain the correct reading order.

Hierarchical evaluation in a hierarchical approach, the previously discussed metrics can be straightforwardly used to assess the results obtained at each level of the hierarchy. The resulting values will help to understand the effectiveness of the method at each level individually.

Furthermore, to compare a hierarchical approach to its flat full-page counterpart, the results obtained for the hierarchical ordering can be flattened into an order of the lowest hierarchy elements (lines in our experiments) at the full-page level. Then, the normalized Spearman's footrule distance can be directly computed as in the non-hierarchical case.

[5]Some datasets may have images that contains two or more pages, in those cases it is common to evaluate all methods at "image level". However, we use the term page level indistinctly as it is the most common case.

On the other hand, since any editing step (swap) in a higher level will update its sub-elements in a lower level as well, the Kendall's Tau rank distance should be simply computed as the sum of the swaps needed at each hierarchical level.

7.1.3 Software

All the experiments described here were performed using the software developed by us to that end based on the PyTorch library[6]. In all cases the software is available online (see Section 8.4), so we hope it will help the research community to replicate our experiments without the time-consuming (and normally underestimated) rewriting task. More importantly, it is very common that many details necessary to replicate the proposed methods and experiments in a given article are not comprehensively described on it, and only the source code will provide all the details.

7.1.4 Hardware

As mentioned before, some statistical models used in this dissertation are very complex and hardware demanding (both in memory and computation resources). For that reason we found essential to use the same hardware configuration in all experiments, so that they are comparable in that sense.

However, in some cases it is possible to improve the performance of a model by allowing it to have access to more computation resources. In any case, we will highlight that restriction when relevant for each experiment.

All experiments were carried on in the same hardware setup, a single NVIDIA GTX-1080 GPU (8GB memory) along with an Intel(R) Core(TM) i3-6100 CPU at 3.70 GHz and 16GB of DDR memory.

7.2 Statistical Models

Most of the experiments described in this chapter have been conducted using ANNs to estimate the different probability distributions specified in the previous chapters. Therefore, we employ different ANN architectures such as MLP, CNN and RPN (see Section 2.3) for each problem. Each proposed architecture is generally described in the following subsections along with main hyperparameters and general details that apply to all experiments.

If necessary, more details will be provided on the respective experiment.

[6] https://pytorch.org

7.2.1 Hyperparameter Selection

Although ANNs are potent models that aim to learn from the data available with minimal human intervention, there are still some hyperparameters that must be tuned manually. For instance, the architecture of the network, the size of each convolutional filter, the number of anchors in an RPN, etc.

To that end, we first use a validation sub-set to define a set of base hyperparameters for each architecture, then we use the whole train set (validation set included) to train the models and report results on the unseen test set. We try to use the same parameters in all datasets for the same experiment in order to avoid an intricate experimental section, helping in that way the clarity and readability of the document. However, slightly better results can be achieved in some cases if the hyperparameters are adjusted for each dataset, model and task individually.

7.2.2 ANN Architecture

As explained in Chapter 4, Chapter 5 and Chapter 6, each of the proposed methods can be implemented by means of an ANN. However, each method have different characteristics to be analyzed in order to design and choose a specific ANN. Hence, in the following sections, we propose a different architecture for each method.

7.2.2.1 Map-Based Approach

As explained in Chapter 5, Baseline Detection and Region Segmentation problems can be handled by the same formulation under the *Map-based* approach (separately or integrated). Consequently, the same proposed ANN architecture will be used in all experiments related to *Map-based* approach.

In this work, we base our network in the well-known UNet architecture [RFB15]. This architecture has shown very good results in several similar problems and is very fast to train [Iso+17]. The UNet architecture is a fully convolutional network that combines a contracting path (encoder) and an upsampling path (decoder), used to capture the context in the image with numerous feature channels between the contracting and upsampling paths. Those feature channels allow the network to propagate context information to higher resolution layers [RFB15]. Moreover, we use the slight modification to the UNet architecture, introduced in [Iso+17], where max-pooling layers on the contracting path were removed, instead, the hyperparameters of the convolutional layers are designed to reduce the size of the feature maps on each layer.

The basic diagram of the Unet-based architecture, used in the experiments of this dissertation, is shown in Figure 7.3. Each block in the encoding path is composed of a convolutional layer, followed by a batch-normalization layer and an activation function. Similarly, each block in the decoding path is composed of

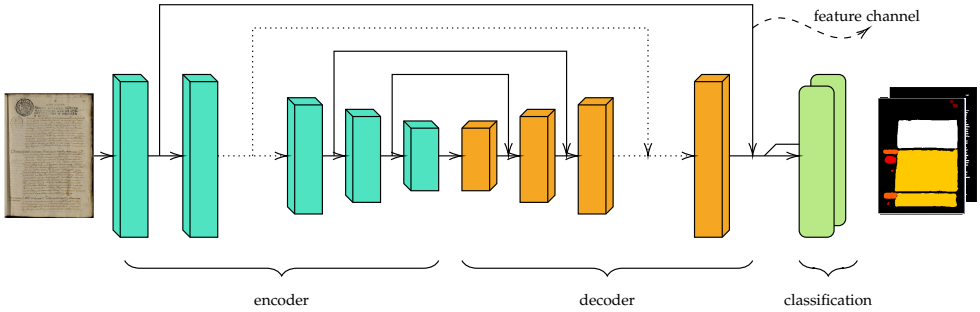


Figure 7.3: Basic diagram of the Unet-based architecture.

a transposed convolutional layer followed by a batch-normalization layer and an activation function. The main parameters of those layers are defined in Table 7.1.

Finally, the classification block varies upon the task to be performed, as it is composed of one output layer for each task (Baseline Detection, Region Segmentation) and the number of filters depends on the number of classes on which classification is performed.

Unless otherwise specified, in all *Map-based* approach experiments, the input images are resized to 1024×768 pixels and the *batch size* is set to eight images per batch in order to comply with the hardware constrains. Also, δ, used to build the map $\mathbf{m}$, is set to 8 pixels (notice that $\mathbf{m}$ has the size of the resized image). Moreover, each model is trained during 200 *epochs*, using mini-batch SGD and Adam solver [KB15], with a learning rate of 0.001, and momentum parameters $\beta_1 = 0.5$ and $\beta_2 = 0.999$.

7.2.2.2 Direct Approach

The *Direct* approach requires and ANN a bit more complex that the *Map-based* approach, because it not only requires to make a prediction at pixel level but also to generate a set of RoIs in the input image where is highly probable that those predictions make sense.

To that end many ANN architectures have been proposed to address similar problems [RF17; Ren+16; He+17]. However, we found Mask-RCNN architecture [He+17] to better meet the requirements of this approach, since it is a multi-stage architecture specifically designed for multi-task problems.

In Figure 7.4 a basic diagram of the architecture is presented. First, a *backbone* network is used to extract a good feature representation of the input data. Then a *Region Proposal Network* is used to generate a set of rectangle boxes which likely contain objects of interest. Next, a *RoI Pool* layer is used to align the predicted

Table 7.1: Architecture of the ANN used in experiments related to the *Map-based* approach, where Conv. stands for Convolution and T. Conv. for Transposed Convolution.

Configuration	Value
Encoder block	
Num. layers	8
Conv. filters	$\{64, 128, 256, 512, 512, 512, 512, 512\}$
Conv. size	4×4
Activation	$\{\{\text{LeakyReLU}\}^7, \text{ReLU}\}$
Batch normalization	$\{no, yes, yes, yes, yes, yes, yes, no\}$
DropOut	*no*
Decoder block	
Num. layers	7
T. Conv. filters	$\{512, 512, 512, 512, 256, 128, 64\}$
T. Conv. size	4×4
Activation	ReLU
Batch normalization	$\{no, yes, yes, yes, yes, yes, yes\}$
DropOut	$\{no, 0.5, 0.5, no, no, no, no\}$
Classification block	
Num. Layers	Num. Tasks
T. Conv. filters	Num. output classes
T. Conv. size	4×4

rectangle boxes between the input image and the feature maps. Finally, a *Prediction Heads* network is used to perform a set of tasks on the objects of interest, for instance classification, bounding box regression and mask generation.

Here we briefly describe Mask-RCNN architecture, while details are presented in Table 7.2.

***Backbone*:** The *backbone* network is composed of a CNN that extracts features from the input image. Commonly, the *backbone* is composed of blocks of ResNet [He+16], VGG [SZ14] or similar architectures that extract features at the scale of the input image. Also, it can be combined with a Feature-Pyramid Network (FPN) [Lin+17] to extract features at different hierarchical scales.

Indeed, in this work we use a combination of 4 ResNet-50 blocks along with 4 FPN hierarchical scales (normally called P2, P3, P4, P5 and P6), as detailed in Table 7.2.

Table 7.2: Architecture of ANN used in the experiments related to the *Direct* approach, where Conv. stands for Convolution and T. Conv. for Transposed Convolution

Configuration	Value
Backbone	
Steam Conv. size	7×7
Steam Conv. stride	$(2, 2)$
ResNet-50 stages	{res2, res3, res4, res5}
FPN levels	$\{P2, P3, P4, P5, P6\}$
FPN lateral Conv. size	1×1
FPN output Conv. size	3×3
RPN	
Num. shared layers	1
Shared Conv. filters	256
Shared Conv. size	3×3
Activation	ReLU
Anchor size	$\{32, 64, 128, 256, 512\}$
Anchor ratio	$[1:1, 1:2, 2:1]$
RoI Pool	
Algorithm	RoIAlign [He+17]
Prediction Heads	
RoIAlign size	7×7
Shared FC. Head layers	2
Shared FC. Head units	$\{1024, 1024\}$
Activation	ReLU
Object Mask Head	
RoIAlign size	7×7
Num. Conv. layers	4
Conv. filters	$\{256, 256, 256, 256\}$
Conv. size	3×3
Num T. Conv. layers	1
T. Conv. filters	256
T. Conv. size	2×2
T. Conv. stride	$(2, 2)$
Activation	ReLU

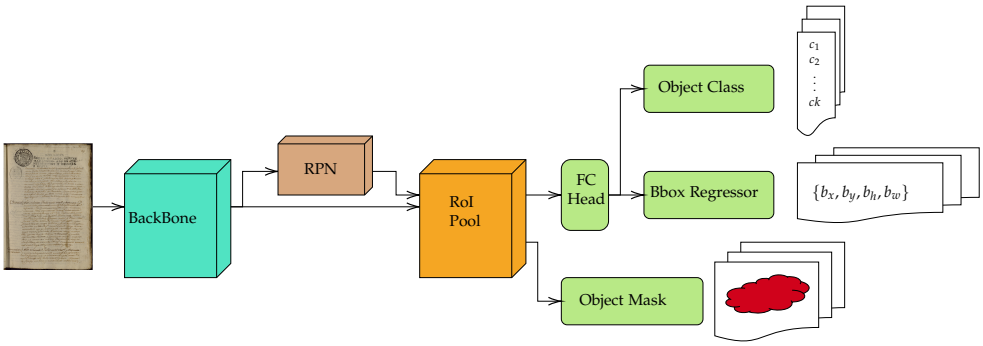


Figure 7.4: Basic diagram of the Mask-RCNN architecture.

Region Proposal Network: It is another convolutional network that scans the feature maps generated by the *backbone* by an sliding window, simultaneously regressing region bounds and scoring the membership of each region to a set of object classes. On each location of the sliding window, k region proposals are predicted, parameterized relative to a set of k reference boxes or anchors. Typically, anchors span a range of sizes and aspect ratios (e.g., 32, 64, 128 and 1:1, 1:2, 2:1).

Then each region proposal is filtered by its scores and a non-maximal suppression algorithm is used to handle overlaps. Finally, the region proposals that remain are called "regions of interest" (RoI) and passed to the next stage.

The details of the RPN are shown in Table 7.2, where it is important to notice that each anchor size is applied to its respective FPN hierarchical scale.

Prediction Heads: also called Prediction Branches, are a set of neural networks (convolutional or not) designed to generate a task specific prediction for each RoI generated on the previous stage. For instance, the most common prediction heads for object segmentation are region labeling (Object class head), region bounding box regression (bbox regression head) and mask prediction (Object mask head).

Region labeling classifies each RoI into a set of "object classes", for instance, in DLA a set of classes could be text-line, paragraph and marginalia. On the other hand, region bounding box regression maps each RoI to an adjusted box that better circumscribe the detected object. This new bounding box could differ from the anchor used to generate the RoI. Finally, the mask prediction head classifies each pixel of the RoI as belonging or not to the silhouette (i.e., the mask) of the detected object.

Unless otherwise specified, in all *Direct* approach experiments, the size of the input images are restricted to 1333 pixels on the longest size and the *batch size* is set to 4 images per batch, in order to comply with the hardware constrains. Also, the

system is restricted to generate up to 1000 RoIs after the non-maximal suppression filter at training time, and no more than 100 at test time. Moreover, each model is optimized during $90,000$ mini-batch SGD iterations, with a learning rate of 0.02 with a multi-step decay of $\gamma = 0.1$ at $60,000$ and $80,000$ iterations.

7.2.2.3 Reading Order MLP

As Reading Order Determination problem complexity can grow rapidly respect to the size of the input document, it is important to keep the model as simple as possible. In [QV21] we estimated this probability using different classifiers, and we found the MLP model to be the most adequate (and faster) for the problem. The main hyperparameters of the model are summarized in Table 7.3 where, as stated, the number of units in the hidden layer varies depending on the size of the input features (as different number of features are used to represent a baseline or a region).

Table 7.3: Architecture of ANN used in the experiments related to the Reading Order Determination problem.

Configuration	Value
MLP	
Hidden layers	1
Units	twice the number of input features
Activation	ReLU

For each element we use a set of features extracted from its layout analysis description, including both geometric and categorical attributes. Specifically:

- Text line (defined by its baseline PLC):
  - The one-hot encoded value of the region type where the text line belongs to.
  - Normalized coordinates of the center of the baseline.
  - Normalized coordinates of the leftmost baseline end.
  - Normalized coordinates of the rightmost baseline end.
- Text Region (defined by its bounding polygon):
  - The one-hot encoded value of the region type.
  - Normalized area of the polygon.
  - Normalized center of mass of the polygon.

– Extreme left, right, top and bottom normalized coordinates of the polygon.

Notice that the memory demands during training could be huge since the number of pairs generated per page is $n^2 - n$, where n is the number of layout elements. In order to reduce the memory footprint, instead of using all possible pairs of the training set, a random set of pairs is generated on the fly directly from the training samples (layout elements) in each epoch. The pairs are composed of the i-th layout element in the training set and another element selected randomly from the same page or region. In this way all the training samples are visited in each epoch, but the memory required is linear with the number of samples.

Each ANN is trained during 3000 epochs, using Adam optimizer with learning rate of 0.001. Also, the batch size is defined to 15000 samples.

7.2.3 Data Augmentation and Pre-trained Models

As mentioned in Section 2.3.1, data augmentation is one of the most common techniques used to circumvent the lack of training data. Based on validation runs, in our experiments we found convenient to use data augmentation in the form of affine transforms (Section 2.1.2.1) and elastic transformations (Section 2.1.2.2) in the case of experiments related to the *Map-based* approach. On the other hand, we found no statistically significant improvement on using it in the case of the *Direct* approach, hence, in this case we do not resort on those data augmentation techniques as it is computationally expensive.

In all the *Map-based* experiments, data augmentation techniques are applied randomly to each sample with 50% probability. Further details about the range of application of each transformation parameter can be found in the source code that accompanies this dissertation.

Respect to Reading Order Determination experiments, the coordinates of each layout element are randomly translated up to 5% from its original position in both vertical and horizontal directions, with a probability of 0.5.

Furthermore, ANN models can benefit from pre-training them on a different task.

With this in mind, we also train and evaluate the aforementioned models by initializing most of the network parameters using external data [GBC16]. To that end, we resort into two different external datasets, namely: ImageNet [Den+09] (composed of natural images) and PubLayNet [ZTY19] (composed of printed documents).

PubLayNet dataset was used to pre-train the models used in *Map-base* and *Direct* approaches (in the PubLayNet task), then prediction layers were replaced by the corresponding prediction layers for the Baseline Detection or Region Segmentation

problem. Finally, the parameters of those prediction layers are initialized randomly and the training procedure re-initialized as in the normal case.

In the case of ImageNet, the dataset is too big to be handled by the hardware defined in Section 7.1.4. Hence, we relay in a public available pre-trained model [7] that uses the same *backbone* network that we employ in the *Direct* approach model. As in the PubLayNet case, the prediction layers are replaced by the corresponding layers of our model. In the case of *Map-base* approach we found no public pre-trained model compatible with our architecture, hence we present no results using that setting.

7.3 Baseline Detection Experiments

As explained in Section 3.3.3 and Chapter 4, Baseline Detection problem aims to obtain the baseline of all lines of text present in a document. Then, those baselines can be used to obtain the text region to feed some ATR system.

In this section we evaluate experimentally the proposed methods described in Chapter 4. First, in Section 7.3.1, the *Map-based* approach and the *Direct* approach are evaluated over OHG and VORAU-253 datasets. We select specifically those datasets in order to evaluate the model not only in a *textual* dataset (which is the common practice) but in a *musical* dataset as well.

Then, in Section 7.3.2, we attempt to establish experimentally how systems respond to limited access to training data. Finally, in Section 7.3.3, we compare our proposed systems with SOTA systems.

7.3.1 Basic Assessment

In this section we assess the proposed methods to address the Baseline Detection problem. All models in this experiment are trained using all the data available for each dataset (i.e., the training set), which is the standard scenario.

We have trained a model for each proposed approach and dataset, and evaluate them on their respective test set. The corresponding F1 values are displayed in Table 7.4 (the complete set of metrics can be consulted in Table C.1).

Using the *Map-based* approach we were able to obtain a 98.04% F1 value in the OHG dataset, which is a very good result that may be used directly by some ATR systems like KWS, where losing a few text lines is still acceptable. Similarly, the *Direct* approach reaches 95.07% F1 value, which is very competitive but suffers from the loss of many baselines (see R value on Table C.1), which makes it less suitable than *Map-based* approach on this task and dataset.

[7] https://dl.fbaipublicfiles.com/detectron2/ImageNetPretrained/MSRA/R-50.pkl

Table 7.4: Baseline Detection F1 value obtained using the proposed methods and different training strategies. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

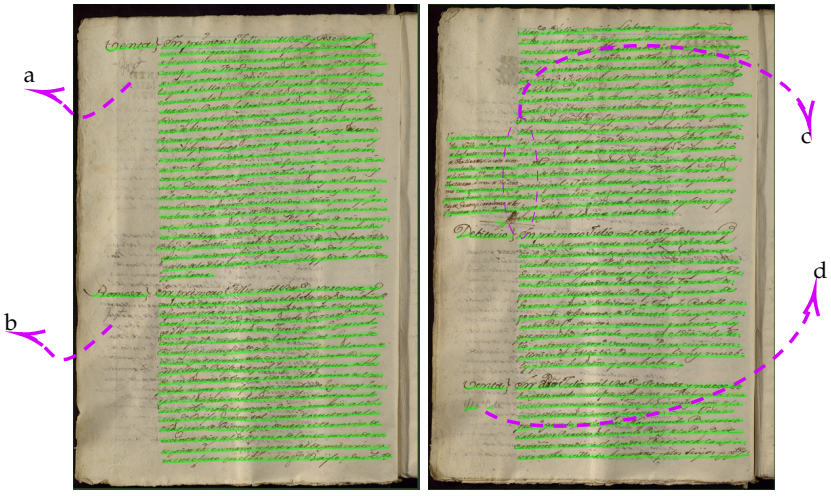
Dataset	method	pre-trained	F1 (%) [CI]
OHG	*Map-based*	—	98.04 [97.82, 98.25]
	Direct	—	95.07 [94.30, 95.77]
	Map-based	PubLayNet	98.29 [98.11, 98.47]
	Direct	PubLayNet	97.02 [96.50, 97.48]
	Direct	ImageNet	97.22 [96.73, 97.66]
VORAU-253	*Map-based*	—	96.22 [95.09, 97.26]
	Direct	—	96.45 [95.28, 97.48]
	Map-based	PubLayNet	95.34 [94.06, 96.52]
	Direct	PubLayNet	96.76 [95.64, 97.77]
	Direct	ImageNet	97.91 [97.07, 98.65]

In the case of VORAU-253 dataset, both models were able to obtain very good results (96.22% and 96.45% respectively). Nonetheless, the difference between them is not statistically significant. It is important to notice that, even though VORAU-253 dataset contains a lot of another layout elements (e.g., staff regions) that can be considered as noise under the Baseline Detection problem, both proposed methods are able to discriminate very well those regions from the text lines.

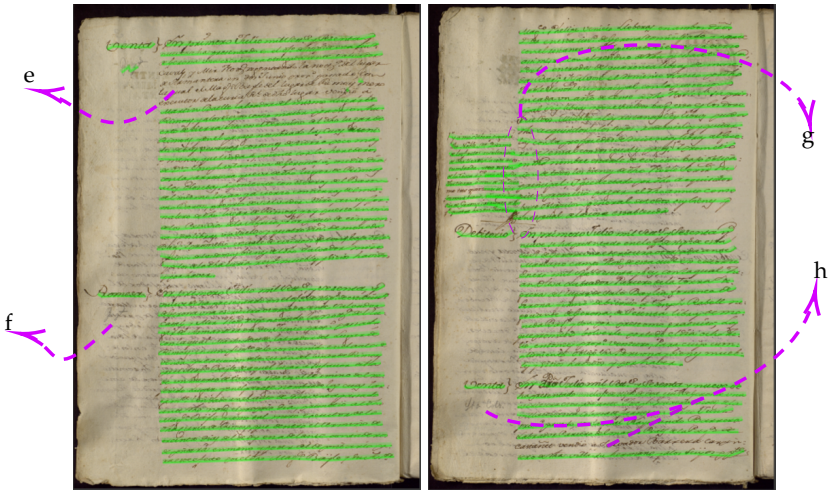
From a qualitative point of view, it is important to highlight that we found hardly any predicted baseline in a background zone. Most baselines were predicted over text regions, as expected, and errors are mainly related to missed, merged or splitted baselines. This observation is important because the predicted baselines probably will be used to feed some ATR system, where any baseline detected on a background region will be a waste of time and resources by the ATR system. Conversely, merged and splitted baselines still useful (although not ideal) for some ATR systems like KWS where the main goal is to be able to search words or other tokens, instead of retrieve the transcript of the document.

In Figure 7.5 and Figure 7.6, examples of the obtained results are shown for OHG and VORAU-253 datasets respectively. In the interest of brevity, we show only two cases that helps us to highlight the most common errors.

In both datasets, merging two or more baselines into a single one is the most common error found using the *Map-based* approach. This kind of errors are due to the fact that $\mathcal{F}(h)$ (Equation (4.2)) is assumed to be bijective, hence any pair of touching or overlapping baselines cannot be fully recovered from the generated map $\overset{*}{\mathbf{m}}$ using Algorithm 1.

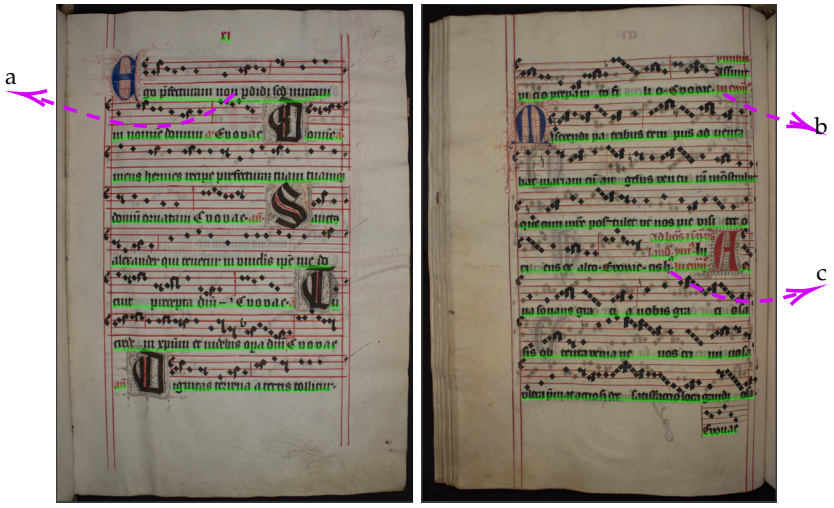


(a) *Map-based*

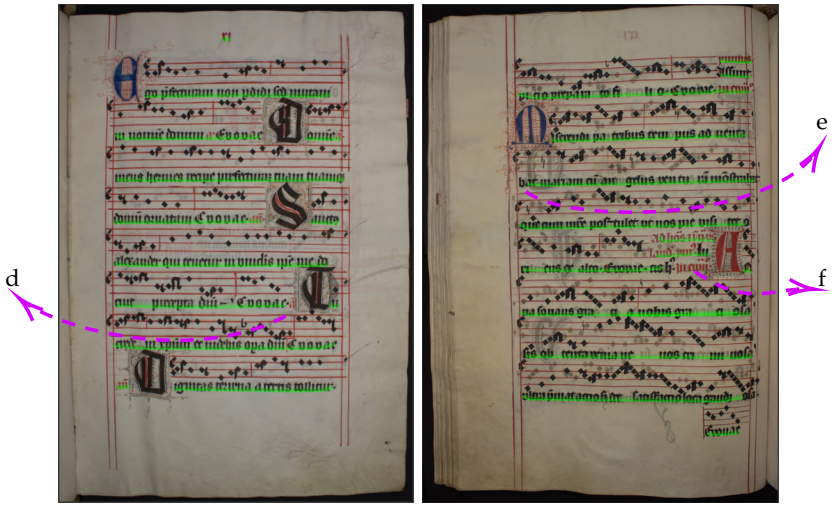


(b) *Direct*

Figure 7.5: Example of Baseline Detection results obtained on the OHG dataset using a) the *Map-based* approach and b) the *Direct* approach. The main errors are highlighted by magenta arrows. Specifically, in the cases "a" and "b" the system missed a baseline, in case "c" the system merged some baselines, and in "d" it split a baseline into two parts. On the other hand, the system missed to predict the baselines in cases "e", "f" and "h". Finally, in case "g" some part of the baselines in the main paragraph extends over the baselines in the marginalia.



(a) *Map-based*



(b) *Direct*

Figure 7.6: Example of Baseline Detection results obtained on the VORAU-253 dataset using a) the *Map-based* approach and b) the *Direct* approach. The main errors are highlighted by magenta arrows. Specifically, in the case "a" the predicted baseline is not well-adjusted to the text, in cases "b" and "c" the system fails by merging two baselines into a single one (in those cases black and red text are different text lines). On the other hand, in the cases "d" and "e" the predicted baselines did not fully cover the text line (missing the last and first letter respectively). Finally, in "f" the system fails to detect a baseline.

Conversely, in the case of the *Direct* approach, it is more common to miss baselines than merging them. As explained in Section 4.1.2, one of the main advantages of the *Direct* approach is to naturally handle touching and overlapping layout elements, making merging errors less common. However, it can fail to generate an RoI that matches some baselines, therefore, completely missing them.

In Table 7.4 we also provide the F1 values obtained using pre-trained models (also, the complete set of metrics is provided in Table C.1).

Pre-trained models strongly increases *Direct* approach results, allowing it to reach an 97.22% F1 value on the OHG dataset (where the difference respect to the *Map-based* approach become very narrow) and even outperforming *Map-based* in the case of VORAU-253 dataset, when pre-trained using ImageNet. Moreover, we found no statistical difference between pre-training the model using ImageNet or PubLayNet data, which is important to be noticed as PubLayNet data is expected to be more related to our task that ImageNet data. Nonetheless, this behavior can be explained by the fact that normaly the first few convolutional layers in a CNN will learn generic filters that help the system to process common characteristics of the input images (corners, lines, etc), hence, a huge dataset as ImageNet will provide a better data distribution to learn those filters.

On the other hand, we found no statistical improvement using a pre-trained model on the *Map-base* approach. Nonetheless, we found pre-trained models more stable across SGD iterations (i.e., loss function decreases smoother than using random initialization).

We attribute this results to the complexity of the models. For instance, the RPN based ANN used in the *Direct* approach is more complex[8] than the ANN used in the *Map-based* approach. Hence, at the beginning of the training process, it is expected to be more difficult to teach a complex model in the direction of the intrinsic distribution of the data, making pre-training a very useful tool to overcome such difficulty.

From the computational cost point of view, the *Direct* approach system requires much more memory[9] and computational resources than the *Map-based* approach. In fact, on average the *Direct* approach takes around 20 times longer to train the system, while, in contrast, the inference time is very similar in both cases.

7.3.2 Effect of Training-data Size

One of the main drawbacks in supervised ML algorithms is, of course, the need of labeled training data to teach the models. DLA is not an exception, and the manual

[8] Here we understood model complexity in terms of the degrees of freedom (i.e., measured in terms of the number of adjustable parameters).

[9] Remember that we have to restrict the batch size to 4 images in this case, while we can use 8 in the *Map-based* approach.

labeling of training data is a very expensive process. Therefore, here we analyze how the size of the training data (i.e., number of available labeled samples) influence the performance of the proposed models. Of course, a model that needs as few training examples as possible is preferable.

To that end we train several models for each dataset, where only a portion of the training data is available to them at training time. We start with as low as 50 samples and increase the training size in batches of 50 samples until we reach all the data available in the dataset.

In the case of OHG dataset we train six different models for each proposed approach (and pre-training setting). However, for the sake of clarity, in Figure 7.7 we only present the F1 value obtained by each model without pre-training as the pre-trained will only improve the results but the tendency is still the same. Nonetheless, the results of all the models and settings can be examined in Figure C.1.

Similarly, in the case of VORAU-253 we train three different models (only 128 training samples are available) for each approach. In the Figure 7.8 we present the F1 value obtained by each model using the same settings described before for OHG dataset. Also, the results of all models and settings can be examined in Figure C.1.

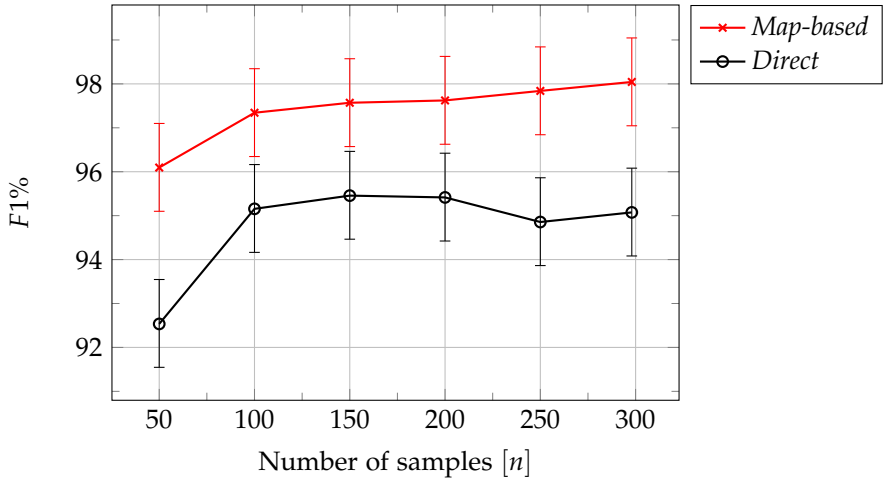


Figure 7.7: Effect of training data size on the F1 value for OHG dataset.

From Figure 7.7, we can observe that both approaches benefit from an increasing number of training samples. Although in most cases the difference is not statistically significant, the tendency is clear to improve. For instance, in both cases the improvement after 100 samples is nominal. Nonetheless, the *Map-based* approach is still outperforming the *Direct* approach in this task for the OHG dataset.

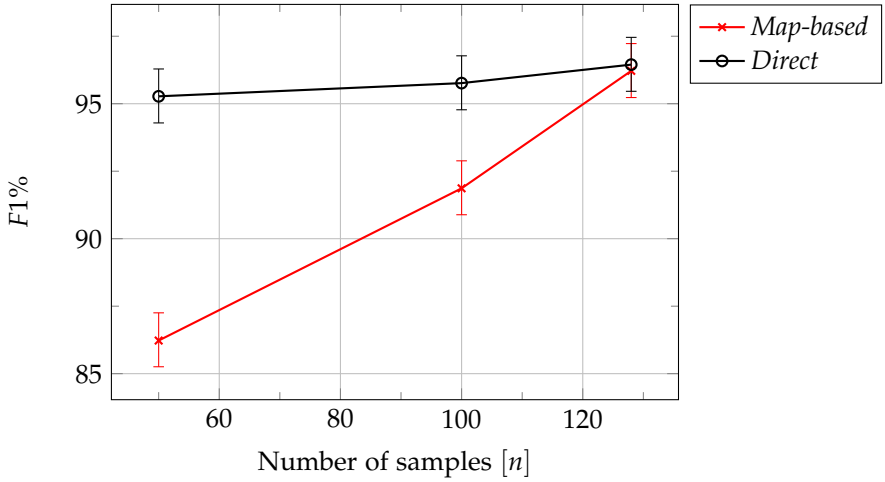


Figure 7.8: Effect of training data size on the F1 value for VORAU-253 dataset.

Conversely, in the case of VORAU-253 dataset (see Figure 7.8), the *Direct* approach did not benefit much from the increasing number of training samples, obtaining almost constant results from as low as 50 pages. On the other hand, the *Map-based* approach starts with an inferior performance ($F1 = 86.2\%$), but it improves up to $F1 = 96.2\%$, catching up with the *Direct* approach. It is important to notice that low results obtaining when training using 50 samples are mainly related to the difficulty of the system to differentiate between normal text and other kind of elements present in the document (remember that VORAU-253 dataset contains textual and musical information, which is a complexity that does not exist in the OHG dataset). For example, as shown in Figure 7.9 the system mistakenly detects four baselines inside the drop-capital letter "S".

Figure 7.9: Example of a common error obtained on VORAU-253 dataset by the *Map-based* approach when trained using only 50 samples. Four baselines were mistakenly detected inside the letter "S".

7.3.3 Comparison with Other Published Works

To compare our systems with other published SOTA works on Baseline Detection, we use three publicly available datasets, namely the Bozens Ratsprotokolle (Bozen), the Competition on Baseline Detection in Archival Documents 2017 (cBAD-17) and the Competition on Baseline Detection in Archival Documents 2019 (cBAD-19) datasets. Further details about the datasets can be found in Appendix B. Nonetheless, it is important to highlight that the Bozen is a common dataset where all pages belong to the same collection, conversely, cBAD-17 and cBAD-19 are heterogeneous datasets composed of a mix of documents from several collections.

We train our models using the default configuration explained on Section 7.2, with the following modifications to address the specific datasets characteristics:

- Bozen: default configuration with no modifications in all cases.
- cBAD-17: in the case of the *Map-based* approach, due to the fact that the dataset contains overcrowded pages with hundreds of text lines where separation between text lines is very small, we increase the size of the input images to 1536×1280 and reduce the hyperparameter δ to 5 (instead of the default 8). As a result, we also have to reduce the batch size to 3 images to complain with hardware constrains. On the other hand, the *Direct* approach is trained using the default configuration with no modification.
- cBAD-19: in the case of the *Map-based* approach we follow the same configuration used for the cBAD-17 dataset. Conversely, in the case of the *Direct* approach, because this dataset contains pages with up to 2452 text lines, the number of RoIs generated by the RPN that survive after the non-maximal suppression filter have been increased to 3000 at training time and 1000 at inference time. Nonetheless, with this configuration the maximum number of baselines to be predicted is only 1000, hence, losing several baselines in such cases where the pages contain more than 1000 baselines. However, increasing those hyperparameters more, heavily impacts the computational resources, hence we are no longer able to train our models under the hardware constrains described on Section 7.1.4. Moreover, we reduce the batch size to only 2 images per batch in consideration of the new parameters and the images size.

In Table 7.5 we report the results obtained by our proposed models, along with SOTA publicly available values.

In the cases of Bozen dataset the *Map-based* approach reaches a 96.82% F1 value using the default configuration, which difference respect to the SOTA value reported by [Grü+19] is not statistically significant. On the other hand, the *Direct* approach is almost 4% below SOTA value. Nonetheless, when including pre-trained models both systems reach SOTA results but again the difference between all of them is

Table 7.5: Baseline Detection results on Bozen, cBAD-17 and cBAD-19 datasets, compared to SOTA values. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Dataset	method	pre-trained	$F1(\%)[CI]$
Bozen	[Grü+19]	—	97.5 [96.6, 97.7]‡
	Map-based	—	96.82 [95.69, 97.84]
	Direct	—	93.83 [91.85, 95.51]
	Map-based	PubLayNet	97.98 [97.16, 98.73]
	Direct	ImageNet	97.91 [97.39, 98.36]
cBAD-17 complex	[Die+17] (winner)	—	85.9
	[ASK18]	—	86.0
	[Grü+19]	—	92.23 [92.14, 92.30]‡
	Map-based	—	87.51 [86.66, 88.33]
	Map-based	PubLayNet	84.67 [83.70, 85.61]
	Direct	ImageNet	88.50 [87.68, 89.29]
cBAD-19	[Die+19] (winner)	—	93.1
	Map-based	—	90.54 [89.70, 91.39]
	Map-based	PubLayNet	89.44 [88.54, 90.32]
	Direct	ImageNet	88.03 [87.17, 88.85]

‡ The number of repetition is not stated in the original publication, hence the confidence interval cannot be fairly compared with ours.

not statistically significant. Of course, as SOTA value is over 97% there is very little room for improvement [10].

On the other hand, when dealing with very heterogeneous datasets like cBAD-17 our systems surpass the $F1$ value obtained by the winner of the competition, without many heuristics and hyperparameter tuning. Despite, better results were reported by [Grü+19] with a similar approach to the *Map-based* system, but using a very strong baseline estimation algorithm instead of our simple $\mathcal{F}^{-1}(\cdot)$ function.

Similarly, in the case of the cBAD-19 dataset, we obtain very competitive results[11], but they are still outperformed by SOTA, ranging from 2.5% to 4.8%. Nonetheless, this experiments highlight one important limitation of the *Direct* approach, which maximum number of predictions is limited by the available computational resources, hence making it difficult to handle documents with hundreds or thousands of layouts elements as in the cBAD-17 and cBAD-19 datasets.

[10] Nonetheless, slightly better results can be obtained by further tuning the network hyperparametres specifically for the dataset, as those reported in [Qui18].

[11] Notice that slightly better results can be obtained by further tuning the network hyperparametres specifically for the dataset, those were reported in [Die+19].

7.3.4 Discussion

In this section we evaluate experimentally how the systems proposed in Chapter 4 respond to address the Baseline Detection problem on handwritten documents.

As seen in the reported results, both proposed methods obtain very good results on textual and musical handwritten documents, where the only input to the system is a digitized version of the page document (i.e., an image). Moreover, both methods rely on probabilistic models to estimate the probability distribution of the input data and make predictions based on it. Remarkably, all the process is carried out with as minimum heuristics as possible, using them only when the computational resources required in finding the optimal solution were unfeasible (see Section 4.1.1.2).

In addition, the main difference between the proposed systems is not its performance but the computational resources required to train the probabilistic models. Indeed, the *Direct* approach requires around 20 times longer to train its ANN and used around twice the memory in the process. Not to mention that, when the training process is complemented with pre-trained models, although no statistically significant improvements were observed, the training process was more stable, which is a very desirable feature.

This experimentation has clarified the impact that the number of training samples has on the overall result of the Baseline Detection task. The results obtained allow us to affirm that, although the global behavior can vary from one dataset to another, in general the tendency is to improve as we add more training samples. Albeit, it is important to analyze on each case when the improvement is too small compared with the effort needed to generate the required ground-truth.

In addition, we compare our methods with SOTA systems on three different datasets. On the standard Bozen dataset our methods reach SOTA results. Conversely, very heterogeneous datasets such as cBAD-17 and cBAD-19 help us to highlight some limitations of out methods, specifically in documents with very crowded text lines or where the number of those text lines is elevated. In those cases, methods such [Grü+19] perform better.

7.4 Region Segmentation Experiments

Region Segmentation is a very important step of DLA. Given an image of a document, the goal is to perform a decomposition of the document image into layout regions (or regions of interest), assigning a relevant label to them in the process.

In this section we evaluate experimentally the proposed methods described in Chapter 5 that aim to address the Region Segmentation problem. Similarly to the previous section, first, in Section 7.4.1, the *Map-based* and *Direct* approaches are evaluated over OHG and VORAU-253 datasets. Then, in Section 7.4.2, we attempt

to establish experimentally how systems respond to limited access to training data. Finally, in Section 7.4.3 we compare our proposed systems with SOTA systems.

7.4.1 Basic Assessment

In this section we assess the proposed methods to address the Region Segmentation problem. All models in this experiment are trained using all the data available for each dataset (i.e., the training set), which is the standard scenario.

We have trained a model for each proposed approach and dataset, and evaluate them on their respective test set. The corresponding mean intersection over union (m^{IoU}), mean Average Precision (AP) and mean Average Recall (AR) values are reported in Table 7.6 (the complete set of metrics can be consulted in Table C.2).

Table 7.6: Region Segmentation results obtained using the proposed methods and different training strategies. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Dataset	method	pre-trained	m^{IoU} (%)$[CI]$	AP	AR
OHG	*Map-based*	—	83.06 [82.16, 83.91]	0.295	0.518
	Direct	—	83.15 [81.95, 84.28]	0.528	0.637
	Map-based	PubLayNet	82.55 [81.70, 83.39]	0.305	0.512
	Direct	PubLayNet	84.43 [83.32, 85.49]	0.573	0.669
	Direct	ImageNet	85.94 [84.98, 86.80]	0.610	0.695
VORAU-253	*Map-based*	—	85.97 [84.60, 87.05]	0.582	0.686
	Direct	—	89.04 [87.79, 89.85]	0.696	0.777
	Map-based	PubLayNet	86.49 [85.42, 87.21]	0.616	0.707
	Direct	PubLayNet	89.39 [88.16, 90.21]	0.714	0.792
	Direct	ImageNet	89.96 [88.74, 90.75]	0.745	0.814

From the point of view of the m^{IoU} metric, both methods perform very similar when no pre-training is performed. Moreover, in the case of OHG dataset the difference between the *Map-based* and *Direct* approaches, without pre-training, is not statistically significant. Conversely, from the point of view of the AP and AR metrics, the systems perform very different, being the *Direct* approach clearly superior.

In effect, this discrepancy between metrics is due to the fact that m^{IoU} is a global metric that does not perform any alignment between layout elements, instead it only analyzes the global value (i.e., label) of each pixel. For instance, in Figure 7.10 and Figure 7.11 we present some cases where the systems fail to predict a perfect layout. Specifically, in the cases "a" and "d" in Figure 7.10a, and cases "b" and "c" in Figure 7.11a, the *Map-based* approach merged two or more layout regions of the same class into a single one. However, m^{IoU} fails to flag that merged region as an

error, as all the pixels in the predicted region belong to the correct class. Instead, AP and AR metrics would take into account that, in the ground-truth, those pixels belong to different layout regions and penalize the merged region accordingly.

On the other hand, the incorporation of pre-trained models help in all cases to improve the systems' performance. Henceforth, *Direct* approach still outperforming the *Map-based* approach on the Region Segmentation problem. In particular, the *Direct* approach pre-trained on ImageNet data reaches a 85.94% m^{IoU} value on OHG dataset along with 0.610 AP and 0.695 AR. Nonetheless, as those metrics suggest, we are still having errors similar to those shown in Figure 7.10b, mainly missing layout regions.

Similarly, in the case of VORAU-253 dataset, the *Direct* approach outperforms the *Map-based* approach by at least 2.9% in m^{IoU} value. In like manner, AP and AR values indicate that the *Direct* approach obtains a much better segmentation of the input document, while as discussed before, the *Map-based* approach performance is limited on documents containing touching or overlapping layout elements.

7.4.2 Effect of Training-data Size

Similar to Section 7.3.2, in this section we aim to experimentally evaluate the impact of the number of training samples in the performance of the proposed models.

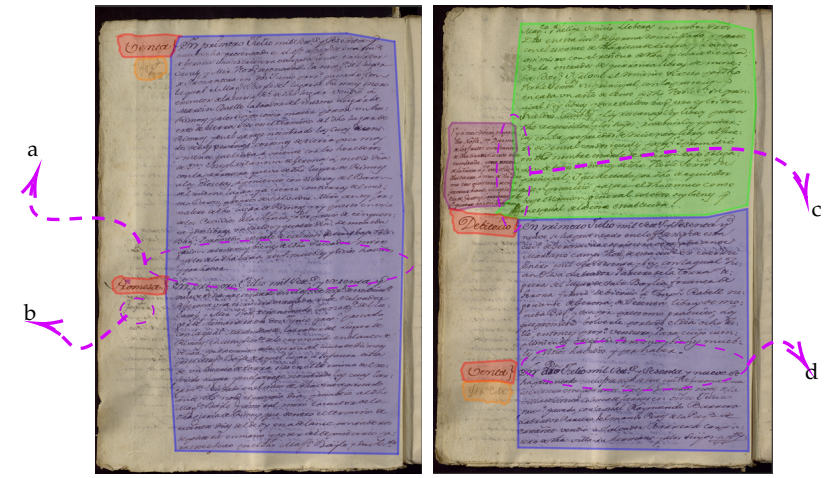
To that end, we train several models for each dataset, where only a portion of the training data is available to them at training time. We start with as low as 50 samples and increase the training size in batches of 50 samples until we reach all the data available in the dataset.

Again, in the case of OHG dataset we train six different models for each proposed approach (and pre-training setting). However, for the sake of clarity, in Figure 7.12 we only present the m^{IoU} and the AP values[12] obtained by each model using the *Map-based* and *Direct* approaches without pre-training. Moreover, all the models and settings can be examined in Figure C.3.

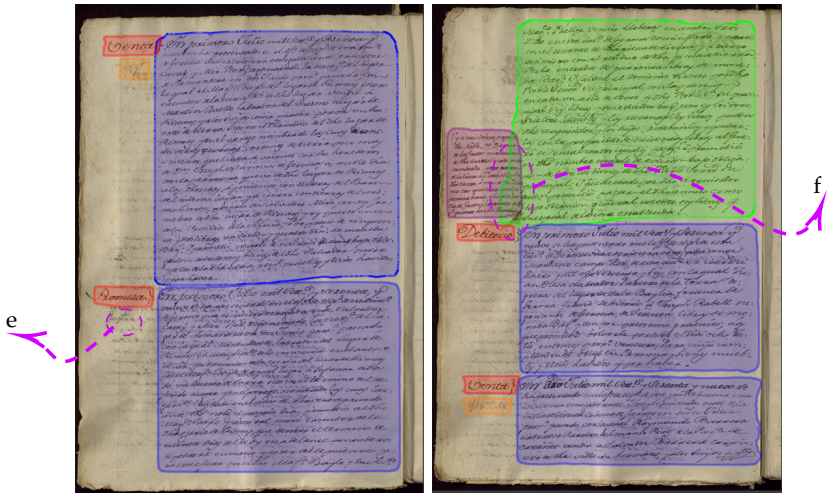
Similarly, in the case of VORAU-253 we train three different models (only 128 training samples are available) for each approach. In Figure 7.13 we present the m^{IoU} and the AP values obtained by each model using the same settings described before for OHG dataset. Furthermore, all the models and settings can be examined in Figure C.4

In both, OHG and VORAU-253 datasets the *Direct* approach outperforms the *Map-based* approach by a large margin (from the AP metric point of view), independently of the number of samples used to train the models. Nonetheless, the tendency on both models is to improve as more training samples were available to them.

[12] AR value follows the same tendency as AP, hence for the sake of clarity we do not present it here.

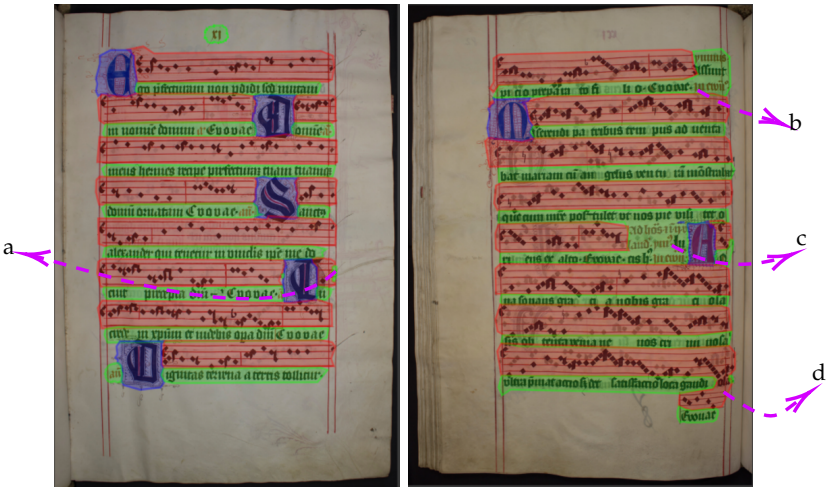


(a) *Map-based*

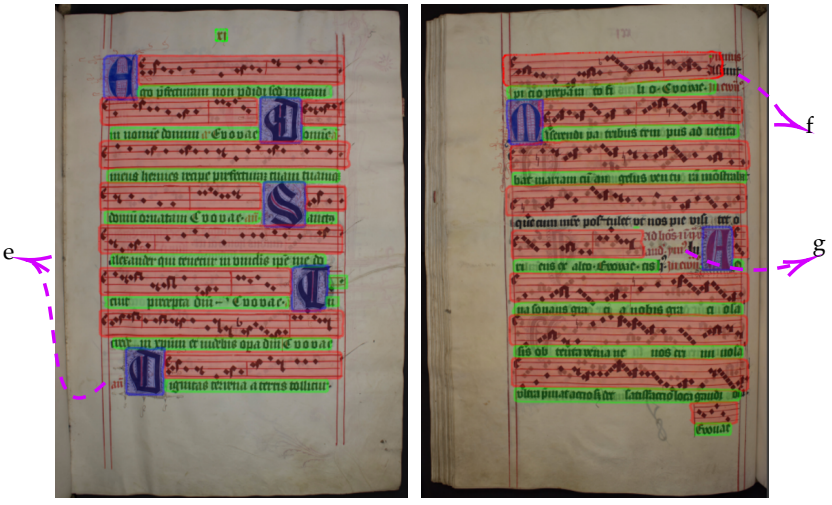


(b) *Direct*

Figure 7.10: Example of Region Segmentation results obtained on the OHG dataset using a) the *Map-based* approach and b) the *Direct* approach. The layout regions are depicted in different colors and the main errors are highlighted by magenta arrows. Specifically, in the cases "a" and "d" the system merged two `par` regions into a single big one, in "b" the system missed a small `nop` region, while in the case "c" the predicted boundary between the `not` and the `pac` regions was not well-defined, so that some text from the `not` region invaded the `pac` region. On the other hand, in the case "e", the system missed a small `nop` region, while in "f", the predicted boundary between the `not` and the `pac` regions was not well defined.



(a) *Map-based*



(b) *Direct*

Figure 7.11: Example of Region Segmentation results obtained on the VORAU-253 dataset using a) the *Map-based* approach and b) the *Direct* approach. The layout regions are depicted in different colors and the main errors are highlighted by magenta arrows. Specifically, in the case "a" the system predicted a spurious region, in cases "b" and "c" the system merged two `lyrics` regions, while in the case "d" merging was between `staff` regions. On the other hand, in the cases "e", "f" and "g" the system missed the respective `lyrics` regions.

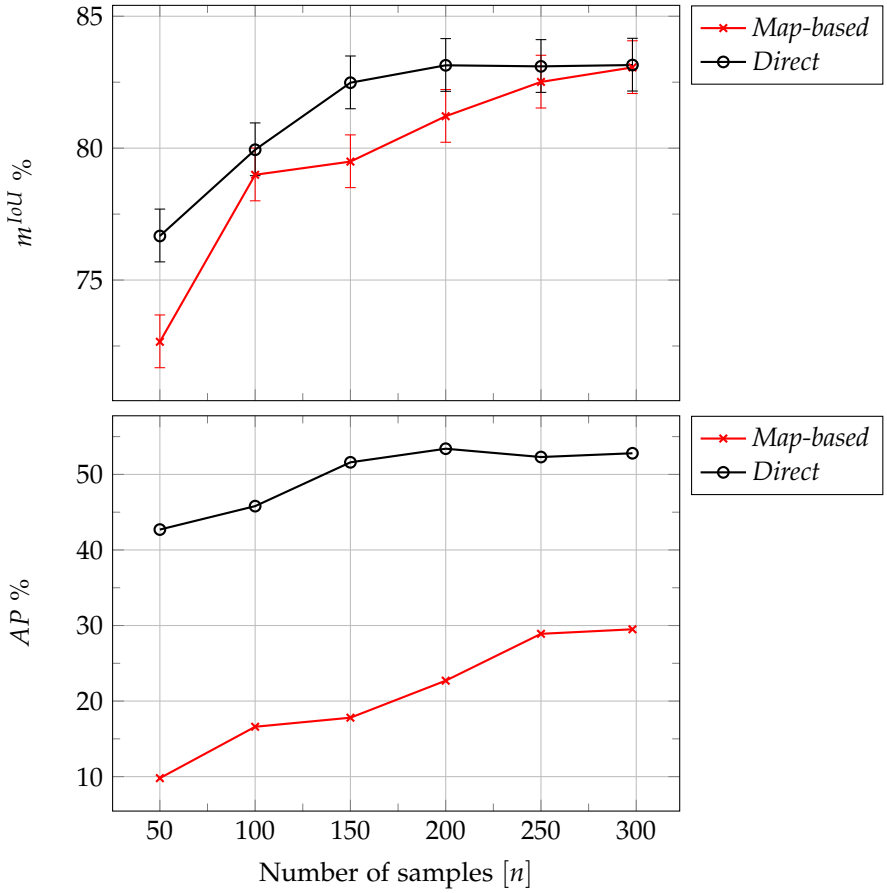


Figure 7.12: Effect of training data size on the Region Segmentation metrics for OHG dataset.

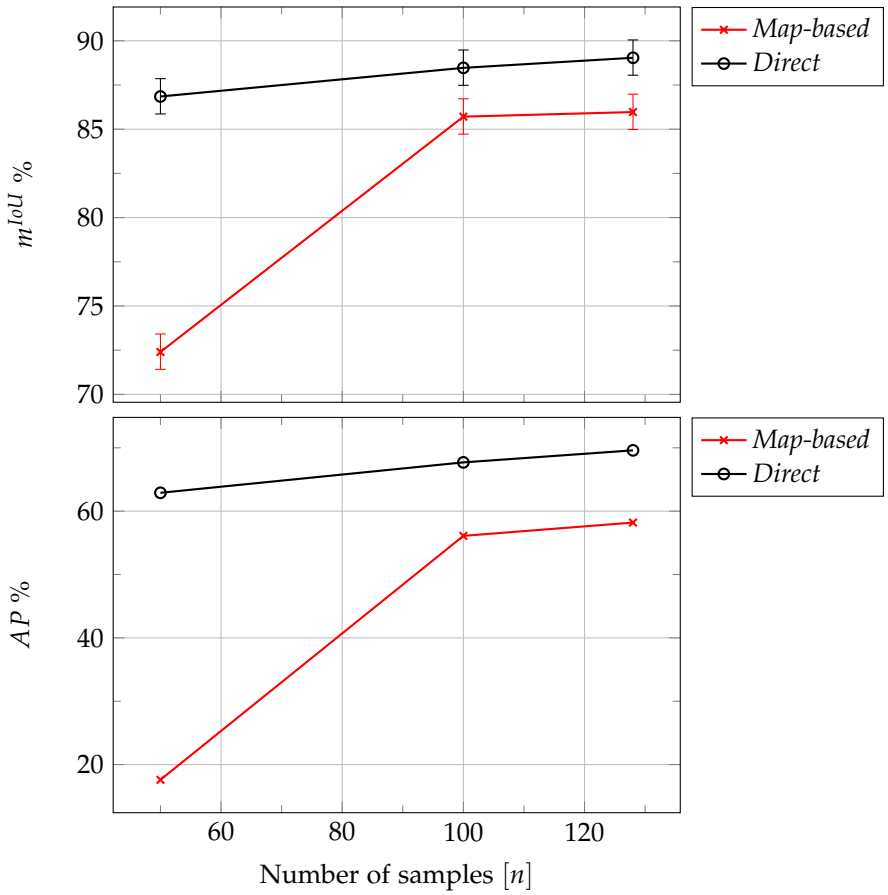


Figure 7.13: Effect of training data size on the Region Segmentation metrics for VORAU-253 dataset.

In the case of *Map-based* approach the improvement is more pronounced, with a global improvement over 10% on m^{IoU} metric on OHG dataset and over 13% on VORAU-253 dataset (same tendency is observed on AP metric).

On the other hand, the improvement observed using the *Direct* approach on the OHG is linear in the first three iterations, and remains constant from that point. Hence, we can obtain very good results in this dataset with only 150 samples.

On the other hand, the improvement observed is very narrow in the VORAU-253 dataset, where very good results can be obtained with as low as 50 training samples. Regardless, the tendency is also to improve as more training samples are available.

7.4.3 Comparison with Other Published Works

There are some datasets labeled for the Region Segmentation problem and systems available for comparison, but most of them are designed for pixel level classification [Sim+16; Sim+17]. Although our proposed systems can perform pixel level classification (actually, they do it as a first step) they are not intended for that task. Hence, in order to perform a fair comparison, we only compare with works were the final goal is to obtain a polygonal representation of the layout regions, not only a pixel level classification.

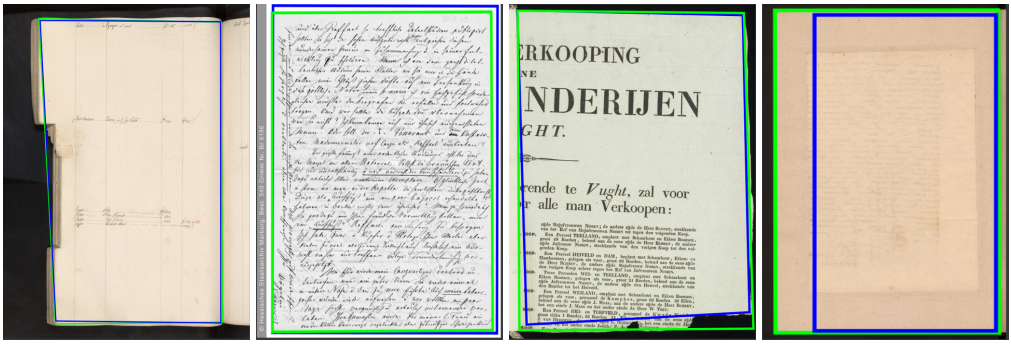
Unfortunately, we were only able to obtain and compare our models using a modified version of the cBAD-17 dataset [Ten+17]. Consequently, we compare our proposed systems with SOTA works published on that dataset using the m^{IoU} metric [13].

The Region Segmentation task defined in [Ten+17] requires to obtain only the four corners of the quadrilateral region that best fit the section of the image that contains the document information (keeping out borders and any other part that does not contain information). However, as we explained in Chapter 5, we found more useful to obtain a polygon that is better suited in situations where the shape of the page cannot be correctly defined by only four vertexes. Nevertheless, in Table 7.7 we directly compare the polygons obtained by our system to the quadrilateral regions defined in the dataset ground-truth, along with the available SOTA methods.

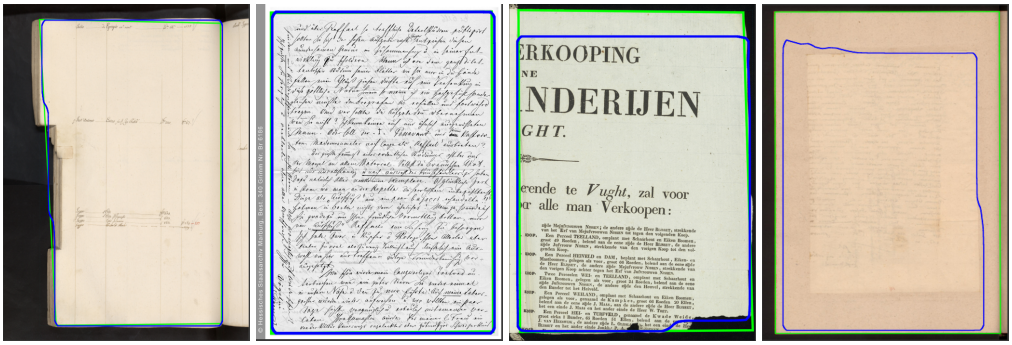
In both cases very good results were obtained on the validation and tests sets. Comparatively, the difference respect to SOTA methods is very small taking into account the characteristics of the m^{IoU} metric. Nonetheless, our methods achieve similar results to human agreement.

Moreover, as depicted in Figure 7.14 most of the errors observed are related to small discrepancies between the proposed region and the ground-truth, and in other cases our systems predict a layout region over-adjusted to the textual information available in the page (e.g., third case in Section 7.4.3).

[13] For the sake of comparability, m^{IoU} values are computed using https://github.com/ctensmeyer/pagenet implementation



(a) *Map-based* apprach. Only small discrepancies are observed in the first two cases. Conversely, last two cases are inaccurate because part of the page is not detected.



(b) *Direct* approach. Only small discrepances are observed in the first two cases. Conversely, in the second and third examples the system predicts a smaller area according to the ground-truth. Nonetheless, all the information in those pages still detected.

Figure 7.14: Representative examples of layouts obtained on the modified cBAD-17 dataset. Green rectangles indicate the ground-truth pages and the blue ones correspond to detections generated by our systems.

Table 7.7: Region Segmentation results obtained on the modified cBAD-17 dataset. Also, SOTA results are presented for comparison. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Dataset	method	m^{IoU} (%)[CI]
cBAD-17* val	[Ten+17] (human agreement)	97.8
	[Ten+17] (automatic)	96.8
	[ASK18]	98.0
	Map-based	97.28 [96.55, 97.80]
	Direct	97.21 [96.50, 97.69]
cBAD-17* test	[Ten+17] (human agreement)	98.3
	[Ten+17] (automatic)	97.4
	[ASK18]	98.0
	Map-based	97.51 [97.18, 97.80]
	Direct	97.14 [96.66, 97.52]

On the other hand, this is, by definition, a very hard problem for the *Direct* approach, because there are many images without any textual information where the system is expected predict a layout region (e.g., last image in Figure 7.14). Remember that the *Direct* approach is trained to differentiate between background and any layout region (see Equation (5.15)), but in those cases (ironically) most of the background is defined as a layout region. Although, under those circumstances, the system performs very well.

7.4.4 Discussion

In this section, we studied the performance of the proposed systems to address the Region Segmentation problem in handwritten documents.

In contrast to the Baseline Segmentation results obtained in the previous section, the results reported in this section suggest a clear superiority of the *Direct* approach over the *Map-based* approach. This superiority is mainly related to the fact that the *Direct* approach can handle touching and overlapping layout regions naturally. Conversely, that is precisely the main limitation of the *Map-based* approach.

Moreover, our results have shown that the *Direct* approach is able to obtain very good results even with as low as 50 training samples, which is very useful as the manual process to generate labeled data to train the models is very expensive. Nonetheless, it is important to take into account that the *Direct* approach is still more expensive in terms of computational resources that the *Map-based* approach.

We also evaluate our methods on the modified cBAD-17 dataset, and compare our results with SOTA methods. Nonetheless, given the very little room for improvement

left by the SOTA results, both proposed methods obtain very competitive results on that Region Segmentation task, reaching very similar m^{IoU} to that of the reported human agreement.

In general, both proposed systems were able to predict very accurate layouts, using only a digitized version of the documents as input. However, our experimental results avail the superiority of the *Direct* approach as it is able to handle touching and overlapping regions.

7.5 Integrated Approach Experiments

Having two different models to address the Baseline Detection and Region Segmentation problems is not completely necessary, what is more, in several cases it is convenient to address both problems in an integrated manner, as explained in Section 5.2.

In this section we evaluate experimentally the effect of addressing both problems in an integrated manner. First, we explore the performance of the integrated models in both tasks using OHG and VORAU-253 datasets. The F1 value and m^{IoU} metric are provided in Table 7.8. Moreover, the results obtained using the specific models for each task is reproduced here as well for easy comparison. Furthermore, the complete set of metrics for all models and settings can be examined in Table C.3 and Table C.4.

In most cases, integrated models performance is not statistically different with respect to the corresponding specific models for each task, but consumes near to half the computational resources. This is an important result that directs us towards a more efficient use of the available computational resources without any statistically significant loss in the systems performance.

Specifically, in the case of OHG dataset, only in the case of *Map-based* approach a significant difference is observed on the Baseline Detection task. However, the difference is narrow (1.78% in the case without pre-training, and 2.21% when pre-trained). Moreover, although that difference in F1 value is significant from the point of view of DLA metrics, as pointed out by [Bos20], its significance is very likely to be drastically lower when measuring the results obtained by a further ATR system.

On the other hand, the difference between results using the integrated and not-integrated approaches on the Region Segmentation task is not statistically significant in all cases. Accordingly, from that point of view an integrated model will be a better option when possible.

7.5.1 Multi-dataset Setting Experiments

Many archives and libraries have a set of "mixed boxes" were several documents are stored without any specific order or classification. It is common to have those boxes

Table 7.8: F1 and m^{IoU} obtained using the proposed methods in an integrated manner on OHG and VORAU-253 datasets. Also, segregated results for Baseline Detection (BD-only column) and Region Segmentation (RS-only column) are reproduced here for easy comparison. Moreover, for simplicity, *-P entries means a model was pre-trained on PublayNet, similarly *-I means it was pre-trained on ImageNet. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	F1 (%)[CI]		m^{IoU} (%)[CI]	
	integrated	BD-only	integrated	RS-only
	OHG			
Map-based	96.26 [95.61, 96.86]	98.04 [97.82, 98.25]	80.54 [79.51, 81.54]	83.06 [82.16, 83.91]
Direct	94.29 [93.44, 95.06]	95.07 [94.30, 95.77]	81.36 [80.05, 82.62]	83.15 [81.95, 84.28]
Map-based-P	96.08 [95.46, 96.65]	98.29 [98.11, 98.47]	81.61 [80.66, 82.53]	82.55 [81.70, 83.39]
Direct-P	97.09 [96.60, 97.51]	97.02 [96.50, 97.48]	83.40 [82.20, 84.55]	84.43 [83.32, 85.49]
Direct-I	96.66 [96.14, 97.12]	97.22 [96.73, 97.66]	85.46 [84.49, 86.37]	85.94 [84.98, 86.80]
	VORAU-253			
Map-based	96.78 [95.65, 97.76]	96.22 [95.09, 97.26]	86.72 [85.63, 87.45]	85.97 [84.60, 87.05]
Direct	95.53 [94.19, 96.72]	96.45 [95.28, 97.48]	88.80 [87.54, 89.68]	89.04 [87.79, 89.85]
Map-based-P	96.30 [95.33, 97.20]	95.34 [94.06, 96.52]	86.65 [85.53, 87.47]	86.49 [85.42, 87.21]
Direct-P	95.94 [94.66, 97.09]	96.76 [95.64, 97.77]	89.69 [88.49, 90.48]	89.39 [88.16, 90.21]
Direct-I	96.01 [94.84, 97.07]	97.91 [97.07, 98.65]	89.99 [88.79, 90.75]	89.96 [88.74, 90.75]

filled with documents that belong to two or more collections, where each collection may have a very different layout. Nonetheless, it is important to process those documents as well, preferably without the time consuming process of classifying them beforehand.

Accordingly, in this section we analyze how the proposed systems perform under a multi-dataset setting. This is, we train our models, in an integrated setting, using samples from dissimilar datasets and analyze how they perform in the respective test set separately.

This setting differs from the common practice to merge several datasets into a single one mainly in two reasons. First, we do not perform any homogenization of the layout, for instance, it is common to group all layout regions with the same kind of content into a single label, however we make no assumption on the content of the document and preserve the original labels of each dataset. Secondly, we evaluate each dataset separately, as we want to know how probable is that the system mixes labels from different datasets (e.g., a sample in dataset "X" is labeled as l_1, but l_1 is a label defined only for dataset "Y").

Henceforth, we use three dissimilar datasets in our experiment, namely OHG, VORAU-253 and Bozen datasets. Those datasets are different enough to force the systems to learn specific probability distributions for each one. Moreover, no label is shared across datasets. Also, datasets with similar content, like a text paragraph in the OHG dataset and a text paragraph in the Bozen dataset, have different labels (e.g., `par` and `paragraph`).

Table 7.9: F1 and m^{IoU} obtained using mixed datasets in an integrated manner on OHG, VORAU-253 and Bozen datasets. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	F1 (%)[CI]	m^{IoU}(%)[CI]	label-accuracy (%)
		Mixed	
Map-based	96.25 [95.75, 96.71]	82.17 [81.45, 82.85]	
Direct-I	96.26 [95.74, 96.73]	84.93 [84.01, 85.82]	
		OHG	
Map-based	96.80 [96.25, 97.28]	80.99 [80.03, 81.91]	99.15
Direct-I	95.85 [95.21, 96.43]	82.84 [81.49, 84.12]	99.89
		VORAU-253	
Map-based	95.82 [94.81, 96.75]	87.54 [86.42, 88.28]	99.97
Direct-I	95.44 [94.25, 96.56]	88.09 [86.88, 89.04]	100
		Bozen	
Map-based	93.75 [91.87, 95.45]	81.66 [80.42, 82.91]	100
Direct-I	95.86 [94.54, 96.95]	86.62 [85.09, 88.04]	100

Based on results presented on Table 7.9, both proposed systems perform very well under the mixed dataset setting. Moreover, comparing this results with the ones reported in Table 7.8 for OHG and VORAU-253 datasets using the integrated approach, we found no statistically significant difference between them, indicating that the mixed dataset approach did not negatively influence the performance on each dataset individually.

Furthermore, in order to analyze the interference between datasets, we count the number of layout regions were the proposed systems correctly classify a layout region into a label that belongs to the dataset where it belongs. We report those results in the column "label-accuracy" in Table 7.9.

Very few errors were found when using the *Map-based* approach. Specifically, in the case of OHG dataset only 24 out of 2864 predicted regions were miss-labeled (i.e., 99.15% accuracy), and in the case of VORAU-253 dataset only 1 out of 4260

layout regions was miss-labeled (i.e., 99.97% accuracy). Finally, in the case of the Bozen dataset we found no miss-labeled regions.

Likewise, using the *Direct* approach we found only 3 miss-labeled regions out of the 2844 predicted in the OHG dataset. Furthermore, we found no errors in the other two datasets.

In the light of those results, one can infer that not only both proposed systems do not decay on a mixed dataset scenario as the one presented in this section. Also, the results obtained can be very useful to perform a further classification of the processed documents into the datasets they belong to (or in a real scenario to sort out the documents mixed in a library box).

7.5.2 Discussion

In this section, we experimentally evaluate how the systems proposed in Chapter 4 and Chapter 5 perform under the integrated and mixed-dataset settings.

Regarding the *Map-based* approach, in most of the evaluated cases we found no statistical difference between the integrated setting and the non-integrated counterpart. Nonetheless, only in the OHG case we found a small deterioration of the F1 value used to measure the performance of the systems in the Baseline Detection problem. Besides, taking into account the benefits of the integrated setting in terms of computational resources and the obtained results, we understand that in practice the *Map-based* approach is very useful alongside the integrated setting.

More patently, results obtained on the *Direct* approach support that the system is able to predict the layout of a document under the integrated setting with the same level of performance obtained in the non-integrated setting, with the benefit of using less computational resources, making it very useful in real scenarios where the computational resources are limited.

Furthermore, we demonstrated empirically the versatility of the *Map-based* and *Direct* approaches in a mixed dataset scenario. We found the proposed systems very useful, not only to address the Baseline Detection and Region Segmentation problems for a specific dataset, but additionally to handle more than one dataset. Consequently, reducing the need of previously sorting mixed collections of documents. Furthermore, the results obtained suggest that the predicted layout can be useful to classify those documents a posteriori.

7.6 Reading Order Determination Experiments

In this section, we experimentally evaluate the performance of the proposed approach to address the Reading Order Determination problem presented in Chapter 6. To that end, first, in Section 7.6.1, we evaluate how the size of the input sample

influence the behavior of the proposed decoding algorithms. Then, the text-lines present in a page document are sorted in reading order in Section 7.6.2 at page level.

Moreover, we analyze how the system performs in a two-level hierarchical setting in order to obtain the text lines in reading order. At the first level, we sort the layout regions in reading order (Section 7.6.3.1). Then at the second level, the text lines inside each layout region are sorted (Section 7.6.3.2). Finally, the reading order obtained in the previously mentioned levels is consolidated at page level in order to be compared with the reading order obtained of the text lines directly at page level (Section 7.6.3.3).

One of the main bottlenecks we found on reading-order-related research is the lack of available datasets. Moreover, none of the datasets used in previous experiments were labeled with the correct reading order, therefore we had to label our own datasets.

We consider three datasets with complex reading order: OHG, Filand Renovated District Court Records (FCR) and READ ABP Table (ABP). FCR dataset is a selection of 500 images (250 for training and 250 for test) containing one or two document pages, and it is annotated at image level using six different region types along with the baselines. This blend of single-page and double-page images is a common complexity added to the DLA problem and the reading problem itself. In Appendix B.6 we provide more details about this dataset.

ABP dataset is a very heterogeneous set of pages where the main element is a table. It is composed of 109 images (55 for training and 54 for test). Only two different regions are defined, namely: "TextRegion" for regular text regions and "TableRegion" for tables. In Appendix B.7 we provide more details about this dataset.

In the case of OHG and FCR datasets we obtain the correct reading order as a byproduct of the transcript and layout already available for it. Conversely, in the case of ABP, since no reading order is explicitly defined in the dataset, we define it as follows:

- First, "TextRegion" elements are sorted in a top-down/left-right manner.
- Then, "TableRegion" cells are ordered row by row from top-to-bottom.
- Finally, inside each cell the baselines are sorted in top-down/left-right order manner.

In Figure 7.15, Figure 7.16 and Figure 7.17, an example of the reading order of each dataset is depicted. As can be noticed, each case is very different from the others and in all cases the reading order is not trivial.

Finally, to easily compare the obtained results, we report the macro average of each metric (see Section 7.1.2.3) relative to the number of test pages or layout regions of each dataset. Additionally, to reduce variance, all reported values have been

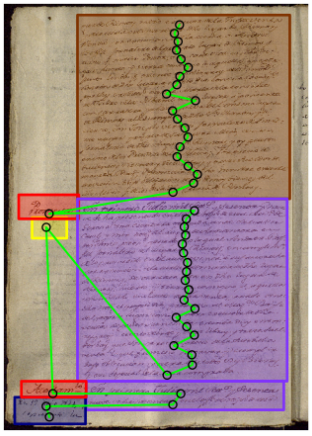


Figure 7.15: Example of the reading order in the OHG dataset. The text-line reading order is depicted in green over the baseline centers, and text regions are depicted as *pac* in brown, *tip* in red, *par* in violet, *nop* in yellow and *not* in blue.

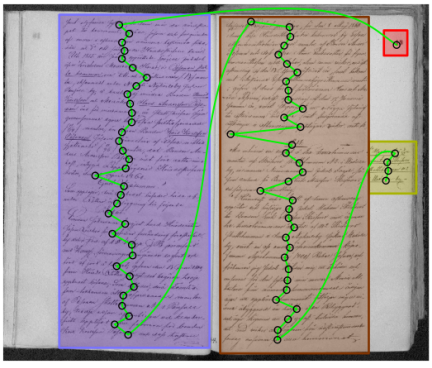


Figure 7.16: A double-page example of the reading order in the FCR dataset. The text-line reading order is depicted in green over the baseline centers, and regions are depicted as *page-number* in red, *marginalia* in yellow, *paragraph* in violet and *paragraph2* in brown.

Figure 7.17: Example of the reading order in the ABP dataset. The text-line reading order is depicted in green over the baseline centers, and the *TextRegions* and *TableRegion* in red and violet respectively.

obtained as the average over ten experiments with different random initialization of the model parameters. Furthermore, $\rho(\mathbf{t}, \boldsymbol{\nu})$ values are reported as a percentage (%) to avoid small numbers.

Along with the proposed approach we also report results using the top-bottom-left-right approach (TBLR) as a basic benchmark in all the experiments. Moreover, a few other methods can be used as a benchmark, but we only use TBLR as it is the only one that does not depend on very specific domain knowledge and heuristics that should be updated for each kind of document.

7.6.1 Decoding Algorithms Analysis

In this experiment we compare the three different decoding algorithms, presented in Section 6.2, for an increasing number of input layout elements. To this end, we build subsets of m text lines per page, from $m = 2$ to the maximum number of lines available (i.e., $2 \leq m \leq n$). The text lines are selected randomly from the OHG dataset (where $n = 40$ on average). We compute the time each algorithm needs to obtain the hypothesis. In each case, the Spearman's footrule distance is computed as well in order to analyze the relative effectiveness of each decoding method. The results are summarized in Figure 7.18.

As shown in Figure 7.18, the Brute Force computing time grows exponentially, hence, we only obtain results for $m \leq 9$ under a reasonable amount of time. In comparison, the computing times of both proposed decoders grow slowly. Nonetheless, the Greedy method typically takes an order of magnitude longer than the FDTD decoder.

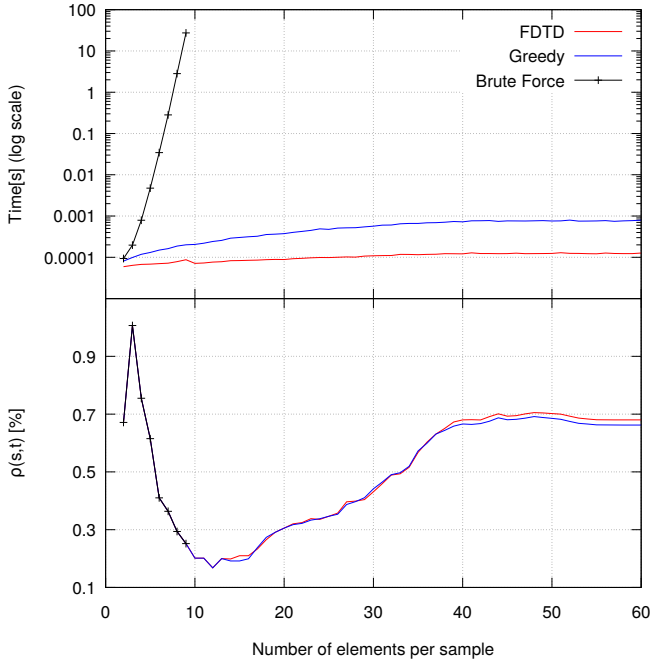


Figure 7.18: Computing time and effectiveness ($\rho(s,t)$) of each decoding method for increasing input sizes.

With respect to the accuracy of each decoder, for $m \leq 9$ all of them provide the same (optimal) response. Then, the Grady and FDTD methods respond very similar until $m > 38$, where the latter starts behaving slightly better. In the light of these results, the FDTD method exhibits better overall performance, taking into account both complexity and accuracy. Nevertheless, for the sake of completeness, in the following experiments, we report results of all decoding algorithms (including Brute Force whenever possible).

7.6.2 Text Lines at Page Level

In this task, we obtain the reading order of the text lines present in a document at page level (i.e., each image is assumed independent of any other in the dataset). In Table 7.10 we summarize the results obtained in all three datasets, using the benchmark method TBLR, and the proposed Greedy and First Decide then Decode (FDTD) algorithms.

Table 7.10: Text-lines ordering results for different metrics and datasets at page level. Reported figures are page averages of values of $\rho(\mathbf{t}, \boldsymbol{\nu})$(in %) and $K(\mathbf{t}, \boldsymbol{\nu})$ (absolute numbers of swaps). Each result is the average over 10 randomly initialized experiments. In both metrics the lower the better.

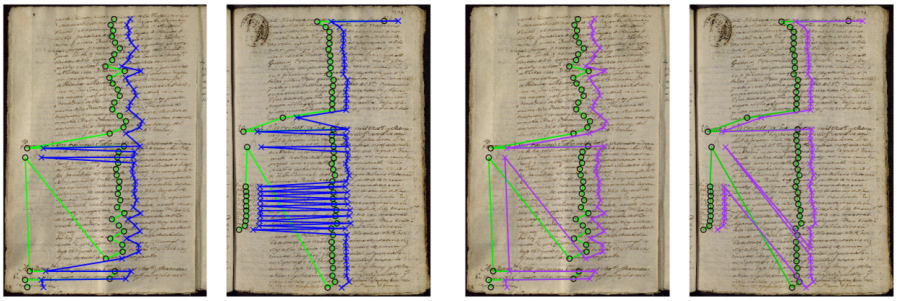
Dataset	Decoder	$\rho(\mathbf{t}, \boldsymbol{\nu})$ (%)	$K(\mathbf{t}, \boldsymbol{\nu})$
OHG	TBLR	2.875	12.899
	Greedy	0.504	2.517
	FDTD	0.498	2.447
FCR	TBLR	31.838	606.976
	Greedy	1.063	18.372
	FDTD	0.699	11.544
ABP	TBLR	10.989	4020.111
	Greedy	4.707	2218.82
	FDTD	4.679	2211.32

Results obtained for the OHG dataset are very encouraging. For the $\rho(\mathbf{t}, \boldsymbol{\nu})$ metric, values below 0.6% are reported for both proposed decoders. This means that even though a few order errors exist, the text lines involved are placed near the correct relative position. In like manner, according to $K(\mathbf{t}, \boldsymbol{\nu})$, less than 3 swaps would be needed by a user to achieve the correct reading order, being FDTD slightly better. Taking into account that OHG dataset contains an average of 40 text lines per page, requiring less than 3 swaps is a very important effort reduction in case any user is required to review and fix the reading order. Moreover, the proposed approach improves TBLR approach results in more than 80%, according to both metrics.

Two representative examples of the reading order obtained in OHG datasets are presented in Figure 7.19. In Section 7.6.2, the result provided by the proposed decoder (FDTD) is fully correct. This is achieved in spite of complexities like the text lines in the middle and the bottom of the page which are very close to each other. In Section 7.6.2, the proposed method produces only one error, in the left marginalia. In contrast, TBLR results are unsatisfactory, particularly in the second example.

It is important to notice that even the single error in the reading order of the second example inserts text from a text region into another region. Such an apparently minor error may dramatically change the meaning of the corresponding paragraphs. This example is a clear indicator of the difficulty and sensitivity of the reading order problem.

The FCR dataset has 64 text lines per image on average, with the added complexity that an image can contain one or two document pages. Above all, the proposed methods obtain a good quality reading order. Specifically, values of $\rho(\mathbf{t}, \boldsymbol{\nu})$ fall below



(a) TBLR method (b) FDTD method

Figure 7.19: Two examples (left-right) of results obtained for the OHG dataset where ground-truth is depicted in green and the system results in blue (for the TBLR approach) and violet (for FDTD). The center of each element is slightly shifted to improve readability.

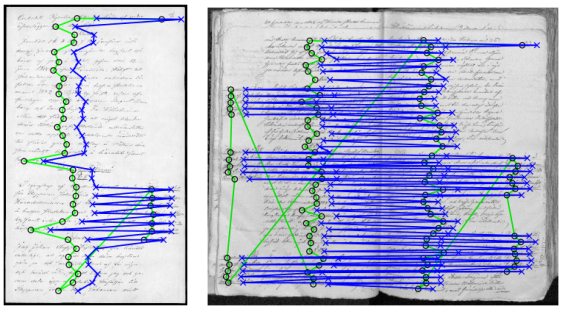
1% of the maximum displacement the elements can have, and, on average, only 18 swaps would be needed to correct the reading order, which is 97% lower than with the TBLR approach. The largest performance differences between the two proposed decoders are observed in this dataset, with FDTD performing better than Greedy by 0.34% in the $\rho(\mathbf{t}, \nu)$ metric and by more than 6 swaps per page according to $K(\mathbf{t}, \nu)$.

Two representative examples of results are shown in Figure 7.20. FDTD rendered almost perfect results (only subtle errors in the last lines of the right marginalia). In contrast, the results of the simple TBLR approach are hardly usable, particularly (as expected) those of the double-page example.

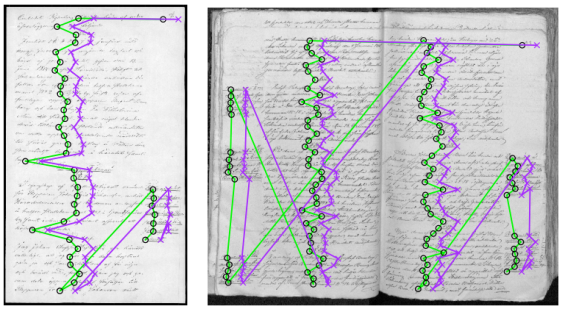
In contrast with OHG and FCR, the ABP dataset is very heterogeneous and only a few samples of each type of table are available for training (see Appendix B.7 for details about this dataset). Moreover, it has as many as 268 text lines on average per page.

Despite the difficulties, the proposed methods are able to obtain reasonably good qualitative results between rows in the same table, while the intra-row results are far insufficient, making the obtained reading order inadequate for real applications. Particularly, the average number of swaps needed to fix the errors is larger than the number of layout elements in the input. Nevertheless, the proposed methods are still performing around 47% better than TBLR according to the $\rho(\mathbf{t}, \nu)$ metric.

Figure 7.21 shows two examples of the reading order obtained on ABP dataset for images that have a sufficiently small number of elements to allow visualization and readability. In both cases, the methods fails to distinguish that text lines that belongs to the same table cell should be ordered before moving to the next cell,



(a) TBLR method



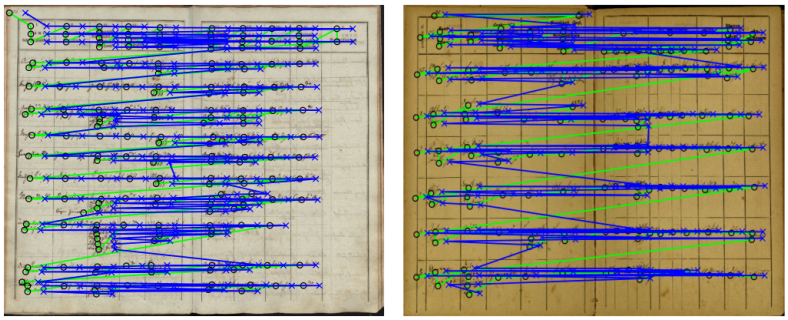
(b) FDTD method

Figure 7.20: Two examples (left-right) of results obtained for the FCR dataset where ground-truth is depicted in green and the system results in blue (for the TBLR approach) and violet (for FDTD). The center of each element is slightly shifted to improve readability.

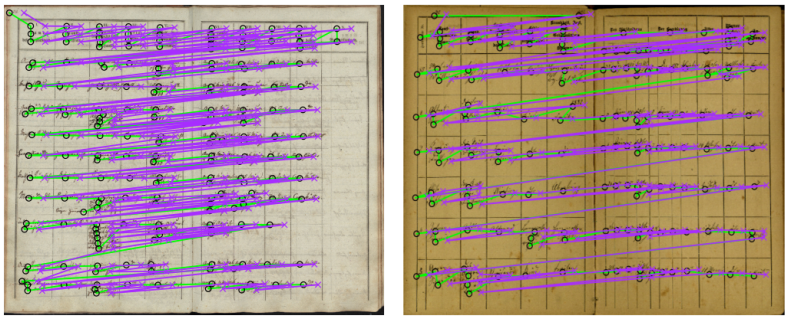
which is a very important limitation of the method.

7.6.3 Hierarchical Approach

In the experiments of this section all the layout regions in the documents are sorted according to the reading order. Then the text lines inside each region are sorted as well. We consider each of these two tasks individually, followed by a hierarchical composition of the individual results at page level. Notice that these experiments, in contraposition to the plain approach (Section 7.6.2), require a DLA system to previously segment the documents into layout regions, not only text lines.



(a) TBLR method



(b) FDTD method

Figure 7.21: Two examples (left-right) of results obtained for the ABP dataset where ground-truth is depicted in green and the system results in blue (for the TBLR approach) and violet (for FDTD). The center of each element is slightly shifted to improve readability.

7.6.3.1 Region Reading Order at Page Level

In this task, we obtain the reading order or the layout regions of each document page. Moreover, as the number of layout regions on the OHG dataset is small enough, in that case, along with the Greedy and FDTD decoders, we provide experiments using the Brute Force (BF) decoder. In this way all the decoders can be empirically compared under reasonable computing resources.

In Table 7.11 we summarize the results obtained in all three datasets. As in the case of text lines at page level, in this experiment, all the proposed decoding methods perform similarly well, and significantly better than TBLR. The difference between the Brute Force method (which guarantees optimal solutions) and the other two proposed methods is minimal. Notice that those results are congruent with the

results reported in Section 7.6.1, proving that the prohibitive optimal solutions are very well approximated by the inexpensive proposed methods.

Table 7.11: Layout regions ordering results for different metrics and datasets at page level. Reported figures are page averages of values of $\rho(\mathbf{t}, \boldsymbol{\nu})$(in %) and $K(\mathbf{t}, \boldsymbol{\nu})$ (absolute numbers of swaps). Each result is the average over 10 randomly initialized experiments. In both metrics the lower the better.

Dataset	Decoder	$\rho(\mathbf{t}, \boldsymbol{\nu})$ (%)	$K(\mathbf{t}, \boldsymbol{\nu})$
OHG	TBLR	9.348	0.614
	Greedy	0.125	0.009
	FDTD	0.134	0.012
	BF	0.118	0.007
FCR	TBLR	28.431	1.860
	Greedy	1.629	0.113
	FDTD	1.692	0.128
ABP	TBLR	16.900	5.148
	Greedy	6.012	0.963
	FDTD	5.920	0.953

Using these methods, the average number of required region swaps is less than 1 every 100 pages for OHG, about 1 each 8 pages for FCR and less than 2 per page for ABP. Equally important is to notice that the Sperman's footrule distance is very small for OHG and FCR, which means that the very few misplaced elements are very near the correct positions.

Despite the unacceptable results obtained for ABP dataset at determining the reading order of the text lines at page level, in this case we obtained very competitive results, mainly because the distribution of layout regions in the dataset is more homogeneous than the distribution of text lines inside each region.

7.6.3.2 Text Lines Reading Order at Region Level

In this section we obtain the reading order of the text lines present in a document page, but restricted to the layout region where they belong. We hypothesized that the reading order problem is simpler to be solved inside each region respect to do it at page level.

Also, in this way we avoid the expensive process of computing $P(r \mid s, s')$ for all $s, s' \in S$ in favor of computing it for only those elements that belong to the same layout region. Later on, this "local" reading order can be consolidated with the region level reading order obtained in the previous section, to obtain the reading order of all text lines at page level.

In Table 7.12 we summarize the results obtained for all three datasets. Notice that in contrast to previous results, in this case all results are normalized respect to the number of layout regions instead of the number of pages.

Table 7.12: Text-lines ordering results for different metrics and datasets at region level. Reported figures are page averages of values of $\rho(\mathbf{t}, \boldsymbol{\nu})$(in %) and $K(\mathbf{t}, \boldsymbol{\nu})$ (absolute numbers of swaps). Each result is the average over 10 randomly initialized experiments. In both metrics the lower the better.

Dataset	Decoder	$\rho(\mathbf{t}, \boldsymbol{\nu})$ (%)	$K(\mathbf{t}, \boldsymbol{\nu})$
OHG	TBLR	0.061	0.013
	Greedy	0.053	0.012
	FDTD	0.053	0.012
FCR	TBLR	0.654	1.859
	Greedy	0.336	1.157
	FDTD	0.338	1.153
ABP	TBLR	0.909	221.111
	Greedy	0.605	114.199
	FDTD	0.603	113.842

In the OHG dataset, writing inside the text regions is very consistent. For this reason, even the results provided by the simple TBLR approach are reasonably good. Yet, the proposed methods still perform slightly better.

Likewise, in the FCR dataset the text line reading order produced by TBLR at region level is fairly good, but the proposed methods achieve a 48% better Spearman's Footrule distance and a 37% lower $K(\mathbf{t}, \boldsymbol{\nu})$ (number of swaps).

With respect to the ABP dataset, the proposed methods are able to reduce the average number of swaps to less than 115 (which is about half of those required by TBLR). However, the number of misplaced elements is still exceedingly large to be useful for real applications.

It is important to notice that we may found layout regions that contain only one text line, hence no reading order is required there. But it may bias the results as the number of those regions in the dataset increases. Hence, in the interest of a fair analysis and comparison, it is preferred to leave that analysis to a consolidation setting, as we do in the next section.

7.6.3.3 Hierarchical Consolidation

As mentioned before, in this section we analyze how the hierarchical approach compares to the non-hierarchical or plain approach. First text regions are sorted at

page level, then text lines within each region are sorted. This allows us to compare this approach with the task of directly sorting text lines at page level.

In Table 7.13 we summarize the results obtained on the hierarchical approach consolidated at page level (called LHP), also the values reported in Table 7.10 are replicated here for easy comparison (column LP).

Table 7.13: Text-lines ordering results for different metrics, at page level and consolidated from the hierarchical approach. Reported figures are page averages of values of $\rho(t, \nu)$ (in %) and $K(t, \nu)$ (absolute numbers of swaps). Tasks correspond to: ordering Lines at Page level (LP) and Lines obtained through Hierarchical processing (LHP). Each result is the average over 10 randomly initialized experiments. In both metrics the lower the better.

Metric		$\rho(\boldsymbol{t}, \boldsymbol{\nu})$ (%)		$K(\boldsymbol{t}, \boldsymbol{\nu})$	
Data	Decoder	LP	LHP	LP	LHP
OHG	TBLR	2.875	3.511	12.899	0.671
	Greedy	0.504	0.035	2.517	0.065
	FDTD	0.498	0.035	2.447	0.069
FCR	TBLR	31.838	31.379	606.976	8.324
	Greedy	1.063	1.074	18.372	4.136
	FDTD	0.699	1.152	11.544	4.138
ABP	TBLR	10.989	8.785	4020.111	1733.092
	Greedy	4.707	4.778	2218.82	893.404
	FDTD	4.679	4.713	2211.32	890.606

In general, the hierarchical approach significantly overcomes the LP results, in some cases by more than one order of magnitude. For instance, in OHG dataset, $K(\mathbf{t}, \boldsymbol{\nu})$ is reduced from less than 2.5 swaps to less than 0.07 swaps, using either the FDTD decoder or the Greedy approach. Important reductions are also achieved for FCR dataset (from more than 18 to a bit more than 4 swaps). Similarly, in the ABP dataset the number of swaps were reduced from more than 2200 to less than 900 swaps.

This important reduction in the number of swaps is due to the fact that a single swap at region level implies moving all its text lines, hence, fixing many errors at once.

Regarding the $\rho(\mathbf{t}, \boldsymbol{\nu})$ metric, results also improve dramatically in the OHG dataset (from 0.5% down to 0.03%) and similarly for FCR dataset. The simple TBLR results are also improved, but the proposed methods still outperform TBLR by large margins.

In summary, the hierarchical treatment of the data proves to be very important both to reduce the problem complexity and to increase the effectiveness of the proposed methods. Although, these improvements come at the price of requiring a richer layout analysis where the text regions are accurately recognized, while in the LP task (text lines at page level) only plain text line detection is strictly required.

7.6.4 Discussion

In this section we experimentally evaluate the proposed approach to handle the Reading Order Determination problem based on learning a pairwise relation operator. Also, two different decoding methods were evaluated experimentally. In general the accuracy of both proposed decoders is very similar, while the main difference is the computational complexity of each one. Experiments support that the FDTD algorithm is faster and slightly more accurate.

We obtain very competitive results in moderately homogeneous datasets such as OHG and FCR, while results in the very heterogeneous ABP dataset are still far away from being useful to recover the information present in the documents.

Furthermore, we evaluate a hierarchical application of the proposed methods, and experimental evidence shows that such a hierarchically processing further reduces the complexity of the problem and increases the accuracy of the results. Nevertheless, the method still exhibits some limitations that should be taken into account in any production scenario. Specially, the method depends on a good estimated $P(r \mid s, s')$, which is proven to be hard in very heterogeneous datasets. Also, the computational cost of samples with a large number of elements (e.g., maritime navigation charts with thousands of elements) should be carefully analyzed.

8 *Conclusions and Perspectives*

This thesis was devoted to the development of probabilistic methods to address the DLA problem on handwritten documents. DLA is the scientific field that aims to extract the intrinsic structure of a document from its digitized version. It includes the detection of the text present in a document, the layout regions and how those elements interact among them to successfully allow any subsequent ATR system to extract the information contained in the data those elements convey. Nonetheless, due to the uncertainty involved in the creation of those documents, it is very complex to decode and extract that non-deterministic structure.

In this thesis we focused on developing probabilistic models to efficiently undertake the uncertainty nature of the problem and predict the most probable structure of each document, while trying to avoid heuristic based methods that included hard-coded knowledge. Therefore, first we address the Baseline Detection problem and propose two probabilistic approaches to the problem. As a matter of fact, the Baseline Detection problem is important in order to feed most SOTA ATR systems developed nowadays.

Next, we focus on the Region Segmentation problem, extending the two probabilistic methods, previously defined for the Baseline Detection problem, to address this new complex problem as well. The resolution of the Region Segmentation problem is helpful to provide better quality digital libraries where, for example, with the help of ATR systems, we will be able to perform text searches as per specific regions in the documents.

Moreover, taking into account real production scenarios where the systems will be deployed, we propose an integrated approach on which both Baseline Detection and Region Segmentation tasks are addressed jointly by the same probabilistic model. This integrated approach helps to resolve the aforementioned problems under limited computational resources without statistically significant degradation in performance.

Finally, we investigate how to address the Reading Order Determination problem, as it is the main vehicle that allows ATR and DLA systems to offer the recognized data as structured information. We propose to address the Reading Order Determination problem as a pairwise probabilistic sorting problem. Furthermore, two different

decoding algorithms have been successfully developed to reduce the complexity of the problem so it can be handled under restricted computational resources.

8.1 Scientific Publications

The different contributions of this thesis have been materialized in scientific publications. Consequently, 6 conference (plus one more pending of submission) and one journal (submitted) paper have been generated. Below we sum up the scientific publications grouped according to the different DLA subproblems they aim to address (notice that some papers convey information about more than one task).

- Baseline Detection on handwritten textual documents: on these papers we address the Baseline Detection problem using the *Map-based* approach on a variety of handwritten documents where the main source of information is textual.
  - Vidal, E., Romero, V., Toselli, A. H., Sánchez, J.A., Bosch, V., **Quirós, L.**, et al. "The Carabela Project and Manuscript Collection: Large-Scale Probabilistic Indexing and Content-based Classification". In: 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR). 2020, pp. 85–90
  - **Quirós, L.**, Bosch, V., Toselli, A. H., and Vidal, E. "From HMMs to RNNs: Computer-assisted Transcription of a Handwritten Notarial Records Collection". In: International Conference on Frontiers in Handwriting Recognition (ICFHR). Aug. 2018, pp. 116–121
- Integrated Baseline Detection and Region Segmentation on handwritten textual documents: on this paper, together with the normal Baseline Detection problem, we addressed the Region Segmentation problem and the integrated approach on textual handwritten documents.
  - **Quirós, L.**, Bosch, V., Toselli, A. H., and Vidal, E. "From HMMs to RNNs: Computer-assisted Transcription of a Handwritten Notarial Records Collection". In: International Conference on Frontiers in Handwriting Recognition (ICFHR). Aug. 2018, pp. 116–121
- Integrated Baseline Detection and Region Segmentation on handwritten musical documents: our previous work was extended to take into account the special requirements of musical handwritten documents. Baseline Detection and Region segmentation tasks are addressed in separated and integrated ways and its performance is evaluated and analyzed.

  - **Quirós, L.**, Toselli, A. H., and Vidal, E. "Multi-task Layout Analysis of Handwritten Musical Scores". In: Iberian Conference on Pattern Recognition and Image Analysis. Springer. 2019, pp. 123–134

- General definition and evaluation of the *Map-based* approach: on this paper we have developed, analyzed and evaluated the main ideas behind the *Map-based* approach. However, we did not publish it yet under the standard channels. Nonetheless, we believe it should be listed here for the sake of completeness.

  - **Quirós, L.** "Multi-Task Handwritten Document Layout Analysis". In: ArXiv e-prints, 1806.08852 (2018). arXiv: 1806.08852

- Reading Order Determination on handwritten textual documents: on this papers we explain the algorithms and methods developed to address the Reading Order Determination problem on handwritten text lines.

  - **Quirós, L.** and Vidal, E. "Learning to Sort Handwritten Text Lines in Reading Order through Estimated Binary Order Relations". In: 2020 25th International Conference on 1405 Pattern Recognition (ICPR). 2021, pp. 7661–7668
  - (Submitted) **Quirós, L.** and Vidal, E. "Reading Order Detection on Handwritten Documents". In: submitted to Neural Computing and Applications Journal (2021).

- DLA confidence and interactive DLA: in this document we did not fully explore the confidence estimation and interactive DLA lines of work, as they are ongoing research lines. Nonetheless, for the sake of completeness, here we list the scientific publications generated during the development of this thesis in those directions.

  - Granell, E., **Quirós, L.**, et al. "Reducing the Human Effort in Text Line Segmentation for Historical Documents". In Proceedings of the 16th International Conference on Document Analysis and Recognition (ICDAR). 2021, pp. 523–537.
  - **Quirós, L.**, Martínez-Hinarejos, C-D., and Vidal, E. "Interactive Layout Detection". In: 8th Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA). Cham: Springer International Publishing, 2017, pp. 161–168.

- General analysis and evaluation of the *Direct* approach: on this paper we have analyzed and evaluated the *Direct* approach to address the Baseline Detection and Region Segmentation tasks (in an integrated manner as well) on textual and musical handwritten documents. Although the paper is not published yet under the normal channels, its preprint can be accessed on arXiv platform.

– **Quirós, L.**, Vidal E. "Evaluation of a Region Proposal Architecture for Multi-task Document Layout Analysis". In: ArXiv e-prints, 2106.11797 (2021). arXiv: 2106.11797.

8.2 Projects and Demonstrators

The methods and algorithms developed in during this thesis have been used in large scale projects, were thousands of images were processed. These projects have served to evaluate the proposed systems in real production scenarios, were unlike the academic ones, the systems should be able to adapt to a dynamic environment were the resources are limited and requirements change easily. Also, production scenarios do not have a labeled ground-truth to measure the performance of the system, only the minimum required to train the system is available (hence the importance of the minimum number of samples needed to train a system). Therefore, no objective metrics can be reported.

Moreover, we developed some demonstrators to showcase the methods capabilities. Those demonstrators have been developed to exhibit an specific capability of the proposed systems, therefore, should not be considered a general production scenario.

Projects:

- *Oficio de Hipotecas de Girona*: the systems developed during this thesis have been tested and furthermore used in a large scale production scenario as part of the transcription process of the *Oficio de Hipotecas de Girona* collection. Initially, an integrated model was used to obtain the baselines and layout regions of the documents using the *Map-based* approach presented in Section 4.1.1. Using this system more than 43 000 images have been processed and then manually revised in order to obtain a ground-truth quality layout. Moreover, in the light of the results obtained using the *Direct* approach we shifted towards this new model and further process more than 10 000 pages.

- Carabela: the *Map-based* approach have been used to process more than 150 000 images related to the "Carabela" project[1]. Due to the high level of degradation of the documents, and the complexity of the layout, Algorithm 1 was updated using some heuristics in order to address the many complexities of the data.

- READ: the software developed to showcase and test the *Map-base* approach (called P2PaLA) has been made part of the Transkribus platform, mainly

[1] http://carabela.prhlt.upv.es/en

developed during the READ EU project[2]. Transkribus is a comprehensive platform for the digitisation, AI-powered text recognition, transcription and searching of historical documents[3].

Demonstrators:

- P2PaLA showroom: on this online demonstrator we show several datasets already processed using the *Map-based* approach presented in Section 4.1.1. The demonstrator is intended to showcase how the proposed system performs in the Baseline Detection task on datasets unseen previously at training time. To that end, we showcase 17 different collections of handwritten documents, comprising more than 6700 images and several layout styles and languages. Particularly, it is important to emphasize that the probabilistic model used to predict those baselines was trained using documents from other collections. The demonstrator can be accessed in the following link:

  http://prhlt-carabela.prhlt.upv.es/tld_showcase/

- Text-line detection demo: on this online demonstrator the user can upload an image of a document and run a Baseline Detection system based on the *Map-based* approach presented in Section 4.1.1. The demonstrator is intended to showcase how the system, previously trained on a set of documents probably unrelated to the image uploaded by the user, performs on an image of an unseen collection. Hence, the demonstrator is designed to process only one image at a time. The system can be accessed in the following link:

  http://prhlt-carabela.prhlt.upv.es/tld/

8.3 Generated Databases

Besides of the common difficulties of developing a thesis, an important fraction of time was dedicated searching for public available handwritten document databases labeled with enough quality to support our experiments. As this thesis is focused not only on one aspect of DLA but three, it was difficult to find a single database labeled to address all problems.

Hence, due to the lack of that public database, we developed new databases (or enhanced a public one) and make them publicly available. We hope those databases will help to boost research on DLA for handwritten documents.

[2] https://www.prhlt.upv.es/wp/project/2016/recognition-and-enrichment-of-archival-documents
[3] https://readcoop.eu

Following, we list all the datasets developed or enhanced during this thesis. Moreover, we refer the reader to Appendix B for further details about each one.

- ***Oficio de Hipotecas de Girona* (OHG)**. This 596 images dataset was created due to the lack of publicly available databases of handwritten text documents annotated at both, baseline and region level, and with a coherent reading order. It is labeled using six different layout regions, the corresponding baselines, the reading order of each element and the line level transcript.

  https://zenodo.org/record/1322666

- **Bozens Ratsprotokolle (Bozen)**. This dataset was originally annotated for HTR, hence, only the transcripts and baselines were available. We re-labeled the 400 images of the dataset to include the different layout regions present on it.

  https://zenodo.org/record/1297399

- **Filand Renovated District Court Records (FCR)**. This 500 images dataset was annotated using six different layout regions, its corresponding baselines and a coherent reading order of the layout elements.

  https://zenodo.org/record/3945088

- **Vorau Abbey library Cod. 253 (VORAU-253)**. As we were interested in covering not only textual documents but also musical ones, we enhanced the VORAU-253 database to include not only the transcript of some staff regions, as originally labeled, but the layout regions and baselines as well. The dataset contains 228 images, now labeled using three different layout regions, along with the baselines present in the "lyrics" of each document.

  https://zenodo.org/record/5443258

8.4 Open Source Software

We do believe that in the new era of Computer Science, it is not enough to share knowledge trough scientific papers. Although they should be the main vehicle to introduce and explain new concepts to the scientific community, papers should be accompanied, whenever possible, by at least a functional implementation of those algorithms and any other source-code used to test them. Provided that, the scientific

community can focus on verify and improving those methods, instead of expending valuable time on the, sometimes, cumbersome process of rewriting code.

Moreover, with the development of very complex Deep Leaning models, which may have hundreds of hyperparameters and many algorithms involved, it is always advantageous to have the source code available in order to analyze and comprehend it. Likewise, it is very common that many important details, necessary to replicate some experiment in a given publication, are not fully described or even mentioned on it, making the process even harder.

Hopping those obstacles do not disturb the results presented in this thesis and our papers, we have decided to release as open source most of the software (at least everything needed to reproduce our experiments) developed during this thesis.

- **Map-based approach to Baseline Detection and Region Segmentation**. We built this deep learning toolkit for DLA to showcase and test the *Map-based* approach described in Section 4.1.1 for the Baseline Detection problem, and in Section 5.1.1 for the Region Segmentation problem. Indeed, all experiments in Section 7.3 to Section 7.5, which are related to the *Map-based* approach were made with this software.

  https://github.com/lquirosd/P2PaLA

- **Direct approach to Baseline Detection and Region Segmentation**. This project is an extension of the *Detectron2* toolkit. We built this project to implement the *Direct* approach described in Section 4.1.2 for the Baseline Detection problem and in Section 5.1.2 for the Region Segmentation problem. Hence, this software has been widely used in our experiments carried out in Section 7.3 to Section 7.5.

  https://github.com/lquirosd/RPN_DLA

- **Reading Order Determination**. The implementation of the algorithms described in Chapter 6, and the code necessary to reproduce all experiments related to the proposed solution to Reading Order Determination (i.e., Section 7.6) can be found in:

  https://github.com/lquirosd/Order_Relation_Operator

8.5 Future Work

As in many other scientific fields, there is a tremendous amount of work still to be done in order to thoroughly address the DLA problem in its full complexity.

Nonetheless, we hope the contributions of this thesis will help in that direction. Moreover, also want to finalize this chapter by identifying the future research topics and development that we have already considered, or we think may be important to the field.

8.5.1 Inter-page Analysis

As mentioned in Chapter 3, the next logical step in DLA is to address the problem at inter-page level. This is, to obtain those structures that exceed the boundary of a single document page in order to fulfill its function. For instance, a chapter in a document must be identified similarly as we identify layout regions in a page, but normally a chapter spans several pages.

During this thesis we assumed that each document page $\mathbf{x} \in \mathcal{X}$ is independent of the others in the collection $\mathcal{X}$. Conversely, in order to address the inter-page analysis task in a probabilistic way, this assumption cannot longer hold.

In order to address this problem, we need to investigate how to extend our algorithms and models to avoid this assumption, at the same time that we do it under restricted computational resources, in a theoretically sound, robust and efficient manner.

8.5.2 Confidence Estimation

Although great progress have been made in the last years in the field of DLA, we are still far away from perfect systems where the expected error on each DLA task is small enough to be ignored in all practical situations (i.e., to consider the problem solved). Therefore, in many cases human intervention is still necessary to review and fix the predicted layout and, of course, to create the ground-truth necessary to train those probabilistic models.

This human intervention is a cumbersome and expensive process that, although it is still necessary, should be abridged as much as possible. Nonetheless, is our belief that estimating the confidence with which a DLA system predicts the layout of some sample, will prove to be very useful to reduce such cumbersome process. For instance, a well-estimated confidence measure can be used to direct the user towards those samples where it is most probable to found an error. Also, confidence measure can be used to reduce the ground-truth generation process load by doing it in an iterative manner, where only a few samples are selected, taking into account the confidence measure, and manually labeled on each step until the model is good enough for a specific application.

Although we have already made some progress in that regard [Gra+21], there is still a lot of work to be done in that direction.

8.5.3 Interactive DLA

In the same direction of reducing the user effort needed to label and review documents for DLA tasks, another approach is to develop an interactive system that assists the user in the task (similar to the CATTI system [Góm10] used in HTR) by means of the Interactive Pattern Recognition framework [TVC11].

To that end, we need to explore how to extend our systems (or develop new ones) to generate several predictions per sample instead of the "best" one that is currently generated. We explore that approach in [Qui+17] using a very simple probabilistic model. However, much work should be done to obtain a theoretically sound system under restricted computational resources.

8.5.4 Probabilistic Indexes Symbiosis

Although it is kind of the "egg and the chicken" problem, it is important to explore the effectiveness of using probabilistic indexes [Pui18] or, if possible, the transcript of the documents, to enhance DLA systems capabilities.

Sometimes the graphical information that can be obtained from the images is not enough for a system to make correct predictions about the structure of a document. For example, instead of searching for the most probable reading order in an image, once knowing what is written on the text lines, we can search for the most probable order restricted to the most probable global transcript of the document.

Nonetheless, normally we need to perform DLA before any ATR system, and we need the ATR system to obtain the probabilistic indexes (or the transcript) that may be used to obtain a better layout. Accordingly, further research is needed in order to solve this "egg and the chicken" dilemma, boosting the benefits of both methods.

8.5.5 Enhance Reading Order Methods

The methods developed during this thesis to address the Reading Order Determination problem, although very competitive, are still underperforming on very heterogeneous datasets, which restricts its applicability to a broach set of production scenarios. Hence, it is convenient to explore more powerful classifiers, as well as more informative features to represent layout elements, including textual content features obtained by means of probabilistic indexing (as mentioned on the previous section).

Equally important is to extend the proposed methods to take into account a more complete context of each layout element. We expect this will lead to more robust ordering models.

Finally, we aim to further exploring the algorithmics of the order decoding problem. According to Equation (6.15), the probability of a solution obtained by the proposed Greedy method is a lower bound of the probability of a globally optimal

solution. Similarly, according to Equation (6.19), the probability of a solution yield by the FDTD method is an upper bound of the optimal probability. Moreover, as shown in Section 7.6, both bounds are fairly tight. These results pave the way for the development of Branch and Bound methods that provide globally optimal solutions as the Brute Force method considered in this work does, but requiring only moderate computation resources.

Appendices

A Numerical Examples on the Reading Order

In this appendix we provide three illustrative examples of possible learned vales of the $\tilde{P}(Y = 1 \mid e_i, e_j), 1 \leq i, j \leq n$ for the five layout elements (text lines) in Figure 6.1 and how Equation (6.8) is computed using these values. These examples also aim to help to understand important aspects of the decoding methods proposed in Section 6.2 and the effect of some corner cases on these methods.

Case 1

In this case we present an example of the most common case, where $\tilde{P}(Y = 1 \mid e_i, e_j)$ is well estimated by some statistical model:

$$\tilde{P}(Y = 1 \mid e_i, e_j): \quad \begin{array}{c} \\ A \\ C \\ E \\ B \\ D \end{array} \begin{array}{c} \begin{array}{ccccc} A & C & E & B & D \end{array} \\ \begin{pmatrix} 0 & 0.6 & 0.9 & 0.8 & 0.7 \\ 0.4 & 0 & 0.8 & 0.6 & 0.9 \\ 0.1 & 0.2 & 0 & 0.7 & 0.9 \\ 0.2 & 0.4 & 0.3 & 0 & 0.8 \\ 0.3 & 0.1 & 0.1 & 0.2 & 0 \end{pmatrix} \end{array} \tag{A.1}$$

We can now use these values to apply Equation (6.8) to the second matrix in Equation (6.3), corresponding to the canonical form of the permutation $\hat{\mathbf{z}} = \{(A, 1), (B, 4), (C, 2), (D, 5), (E, 3)\}$:

$$
\begin{aligned}
P(R^{\hat{\mathbf{z}}}) \approx\ & \tilde{P}(Y=1 \mid A,C)^2\ \tilde{P}(Y=1 \mid A,E)^2 \\
& \tilde{P}(Y=1 \mid A,B)^2\ \tilde{P}(Y=1 \mid A,D)^2 \\
& \tilde{P}(Y=1 \mid C,E)^2\ \tilde{P}(Y=1 \mid C,B)^2 \\
& \tilde{P}(Y=1 \mid C,D)^2\ \tilde{P}(Y=1 \mid E,B)^2 \\
& \tilde{P}(Y=1 \mid E,D)^2\ \tilde{P}(Y=1 \mid B,D)^2 \\
=\ & 0.6^2 \cdot 0.9^2 \cdot 0.8^2 \cdot 0.7^2 \cdot \\
& 0.8^2 \cdot 0.6^2 \cdot 0.9^2 \cdot \\
& 0.7^2 \cdot 0.9^2 \\
& 0.8^2 \\
\approx\ & 0.0043
\end{aligned}
$$

The same result is obtained for the first matrix in Equation (6.3), but using the corresponding values of $\tilde{P}(Y = r_{i,j} \mid e_i, e_j)$ as:

$$
\begin{aligned}
P(R^{\hat{\mathbf{z}}}) \approx\ & 0.8^2 \cdot 0.6^2 \cdot 0.7^2 \cdot 0.9^2 \cdot \\
& (1-0.4)^2 \cdot 0.8^2 \cdot (1-0.3)^2 \cdot \\
& 0.9^2 \cdot 0.8^2 \cdot \\
& (1-0.1)^2 \\
\approx\ & 0.0043
\end{aligned}
$$

And now for the naive TBLR order $\mathbf{z}' = \{(A,1),(B,2),(C,3),(D,4),(E,5)\}$:

$$
\begin{aligned}
P(R^{\mathbf{z}'}) \approx\ & 0.8^2 \cdot 0.6^2 \cdot 0.7^2 \cdot 0.9^2 \cdot \\
& 0.4^2 \cdot 0.8^2 \cdot 0.3^2 \cdot \\
& 0.9^2 \cdot 0.8^2 \cdot \\
& 0.1^2 \\
\approx\ & 0.00000436
\end{aligned}
$$

The Brute Force decoding of Section 6.2 goes over the 120 different permutations to obtain that the most probable permutation is $\mathbf{z}^\star = \hat{\mathbf{z}}$, with $P(R^{\mathbf{z}^\star}) = 0.0043$. The Greedy and FDTD decoders also predict the same permutation.

Case 2

Another possible case, where $\tilde{P}(Y = 1 \mid e_i, e_j)$ is properly estimated (in the sense discussed in Section 6.1) could be:

$$\tilde{P}(Y = 1 \mid e_i, e_j): \quad \begin{array}{c} \\ A \\ C \\ E \\ B \\ D \end{array} \begin{array}{c} \begin{array}{ccccc} A & C & E & B & D \end{array} \\ \begin{pmatrix} 0 & 0.6 & 0.8 & 0.8 & 0.7 \\ 0.4 & 0 & 0.9 & 0.9 & 0.9 \\ 0.2 & 0.1 & 0 & 0.7 & 0.7 \\ 0.2 & 0.1 & 0.3 & 0 & 0.9 \\ 0.3 & 0.1 & 0.3 & 0.1 & 0 \end{pmatrix} \end{array} \tag{A.2}$$

Following Algorithm 4, the most probable element to be placed in the first position is C with $b = 0.2916$ (see lines $6 - 9$ in Algorithm 4). Upon termination, Algorithm 4 finally predicts the reading order as $\tilde{\mathbf{z}}^\star = \{(C,1),(A,2),(E,3),(B,4),(D,5)\}$ with $P(R^{\tilde{\mathbf{z}}^\star}) = 0.003$ (which in fact is the second best permutation), while the global optimum permutation is $\mathbf{z}^\star = \{(A,1),(C,2),(E,3),(B,4),(D,5)\}$ with a higher probability, $P(R^{\mathbf{z}^\star}) = 0.0074$. The same optimal results is also obtained by the FDTD decoder in this case.

This is a representative example of the limitations of the Greedy decoder, where a local maximum corresponds to a sub-optimal result.

Case 3

As discussed in Section 6.2.2, it is possible that Equation (6.17) leads to ties in the number of zeros per row in $R^\star$, resulting in an improper permutation. This may happen, for instance, in the following case, where $\tilde{P}(Y = 1 \mid B, C) > 0.5$, $\tilde{P}(Y = 1 \mid C, E) > 0.5$ and $\tilde{P}(Y = 1 \mid E, B) > 0.5$:

$$\tilde{P}(Y = 1 \mid s_i, s_j): \quad \begin{array}{c} \\ A \\ C \\ E \\ B \\ D \end{array} \begin{array}{c} \begin{array}{ccccc} A & C & E & B & D \end{array} \\ \begin{pmatrix} 0 & 0.6 & 0.9 & 0.8 & 0.7 \\ 0.4 & 0 & 0.8 & 0.4 & 0.9 \\ 0.1 & 0.2 & 0 & 0.7 & 0.9 \\ 0.2 & 0.6 & 0.3 & 0 & 0.8 \\ 0.3 & 0.1 & 0.1 & 0.2 & 0 \end{pmatrix} \end{array} \tag{A.3}$$

This combination contradicts the transitivity property of a total order and, using Equation (6.17) and Equation (6.18), FDTD yields an *improper* permutation; namely: $\mathbf{z} = \{(A,1),(C,3),(E,3),(B,3),(D,5)\}$, with $P(R^\star) = 0.0043$ obtained using (6.8). The optimal solution provided by Brute Force (and in this case also by the Greedy method) is $\mathbf{z}^\star = \{(A,1),(C,2),(E,3),(B,4),(D,5)\}$, with $P(R^{\mathbf{z}^\star}) = 0.0019 < 0.0043 = P(R^\star)$. Depending on how the ties for C, E, B are resolved, different *proper* permutations can be obtained. Nonetheless, only if those ties are resolved in favor of $C \prec E \prec B$, the optimal permutation, $\mathbf{z}^\star$, is obtained.

B Databases

Several datasets were used and processed during the development of this thesis. However, many of them are private or the ground-truth is not completely available, which makes the process of replication or validating results very difficult. Therefore, in order to facilitate the replication or validation of the results presented in this thesis, we report results only using public available datasets.

In this appendix we provide a description of seven datasets used in this document, providing in all cases a link to access that publicly available data.

B.1 *Oficio de Hipotecas de Girona* (OHG)

We create the Oficio de Hipotecas de Girona (OHG) database due to the lack of a publicly available database of handwritten text documents annotated at both baseline and region level and with a coherent reading order.

The manuscript *Oficio de Hipotecas de Girona* (OHG) is provided by the Centre de Recerca d'Història Rural from the Universitat de Girona (CRHR)[1]. This collection is composed of hundreds of thousands of notarial deeds from the XVIII-XIX century. Sales, redemption of censuses, inheritance and matrimonial chapters are among the most common documentary typologies in the collection. This collection is divided in batches of 50 pages each, digitized at 300ppi in 24 bit RGB color, available as TIF images along with their respective ground-truth layout in PAGE XML format, compiled by the HTR group of the PRHLT[2] center and CRHR.

The publicly available database [Qui+18b] is a portion of 596 pages from the collection, from batch *b001* to batch *b012*. OHG pages exhibit a relatively complex layout, as shown in Figure B.1, composed of six relevant region types; namely: `pag`, `tip`, `par`, `pac`, `not`, `nop`, as described in Table B.1. Moreover, it contains only textual information, without any drawing or table.

Nonetheless, there are several details that make the dataset complex from the DLA point of view. For instance, a page can contain two or more `par` regions with very narrow or even no separation between them (see `par` (blue) regions in the left

[1] http://www2.udg.edu/tabid/11296/Default.aspx
[2] https://prhlt.upv.es

Table B.1: Layout regions in the OHG database.

ID	Description
`pag`	a page number.
`tip`	a notarial typology.
`par`	a paragraph of text that begins next to a notarial typology.
`pac`	a paragraph that begins on a previous page.
`not`	a marginal note.
`nop`	a marginal note added a posteriori to the document.

page in Figure B.1), or it can contain layout regions of different class but one of them is embedded over the other (like `pag` and `pac` regions on left page in Figure B.1) which makes impossible to fully segment those regions using only rectangles (i.e., at least a polygon with more than four vertexes is needed).

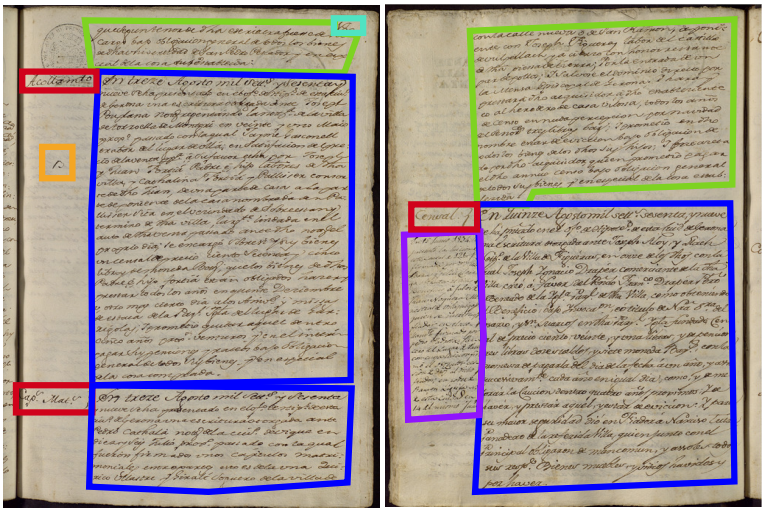


Figure B.1: Examples of pages with different layouts, belonging to the OHG database. Cyan: `pag`, red: `tip`, green: `pac`, blue: `par`, violet: `not`, orange:`nop`.

Another important complexity of the OHG dataset, concerning the Baseline Detection problem, are the baselines that belong to different text regions but the physical separation between them is minuscule, so separating them is a very hard task. For example, in the right page in Figure B.1 there is virtually no separation between baselines belonging to the `not` region (in violet) and the ones belonging to the `par` region (in blue).

Equally important is to analyze how the layout regions labels are defined. In OHG we have `pac` and `par` regions, of which the only way to differentiate them is that if there exists a `tip` region on the right side of the layout region, it should be labeled as `par`, otherwise it should be labeled as a `pac` region. For example, in the case of the first paragraph of the right page in Figure B.1 there is no `tip` region on its right, hence it is labeled as a `pac` region. Provided that, any DLA system designed to address the Region Segmentation task, must be able to take into account not only the local information of some crop of the image, but a more global context.

Furthermore, in Table B.2 the main statistics of the OHG dataset are listed. From which, we can notice that in average each page contains 39.9 [33, 66] lines of text (hence same number of baselines), distributed over and average of 4.8 [1, 11] layout regions per page.

Table B.2: Main characteristics of the OHG database.

Region	#Regions			#Lines		
Name	Train	Test	Total	Train	Test	Total
`pag`	157	139	296	157	139	296
`tip`	430	421	851	458	455	913
`par`	430	421	851	7562	7333	14895
`pac`	240	238	478	3519	3561	7080
`not`	17	17	34	95	98	193
`nop`	210	191	401	216	191	407

Finally, the dataset can be downloaded from https://zenodo.org/record/1322666.

B.2 Vorau Abbey library Cod. 253 (VORAU-253)

VORAU-253 is a music manuscript referred to as Cod. 253 of the Vorau Abbey library, which was provided by the Austrian Academy of Sciences. It is written in German Gothic notation and dated around year 1450. This manuscript is interesting because of the complexity of its layout, where staff, text and decorations are intertwined to compose the structure of the document (see Figure B.2).

This database is a subset of 228 pages of the archive, using 128 randomly selected pages for training and 100 for test.

We annotate this database manually into the following three layout regions:

- `staff`: represents the regions that contains a set of horizontal lines and spaces where each one represent a different musical pitch. This region type does not contain text lines. Hence, no baselines.

- `lyrics`: they are the words that are sung, appearing below their corresponding staff, and other text in the document. In all cases, text to be sung and the other text are assigned to different layout regions under the `lyrics` label.
- `drop-capital`: is a decorated letter that might appear at the beginning of a word or text line. As it is a single big letter, it contains no text lines nor baselines.

An example of this layout is shown in Figure B.2. It is important to notice that the decorations that accompany some `drop-capital` letters are not labeled, as the main interest is in the letter itself rather than in the decoration.

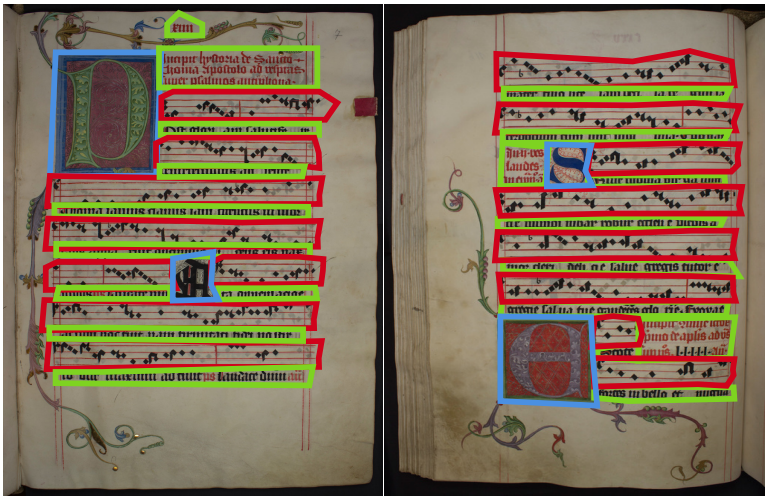


Figure B.2: Examples of pages with different layouts, belonging to the VORAU-253 database. Blue: `drop-capital`, red: `staff`, green: `lyrics`.

Similarly to the OHG dataset, on VORAU-253 we can found several cases were the sole geometrical information of a layout region is not enough to correctly label it, hence, a more global context is necessary. For instance, in the case of the `lyrics` regions, it may happen that a `lyrics` region with text to be sung is next to other `lyrics` region but with text that is not part of the song itself. In those cases the only way to define the border between them is to take into account the context of the layout region (e.g., a `lyrics` region with text to be sung must be placed below to a `staff` region) and probably its meaning (i.e., if the text should be sung or not).

This kind of complexities, along with the fact that the document contains musical information instead of only textual data, makes this dataset very interesting for the DLA problem. The main statistics of the database are presented in Table B.3,

from which we can obtain that on average each page contains 12.5 [7, 23] text lines distributed over an average of 10.5 [7, 15] `lyrics` regions. Moreover, each page contains 22.3 [14, 28] layout regions on average.

Table B.3: Main characteristics of the VORAU-253 dataset.

Region	#Regions			#Lines		
Name	Train	Test	Total	Train	Test	Total
drop-capital	336	232	568	—	—	—
staff	1194	919	2113	—	—	—
lyrics	1379	1042	2403	1628	1215	2843

This dataset can be downloaded from https://zenodo.org/record/5443258.

B.3 Bozens Ratsprotokolle (Bozen)

This database consists of a subset of documents from the Ratsprotokolle collection composed of minutes of the council meetings held from 1470 to 1805 (about 30.000 pages)[Tos+18]. The database text is written in Early Modern German by an unknown number of writers. The public database is composed of 400 pages (350 for training and 50 for validation); most of the pages consist of a two or three zones with many difficulties for line detection and extraction.

The ground-truth of this database was available in PAGE format and annotated at baseline level. Furthermore, we manually update the ground-truth to include the regions of the documents and made it publicly available as well.

The layout regions have been labeled using four different, self-explanatory, region types, namely: `page-number`, `marginalia`, `heading`, and `paragraph`. In Figure B.3 an example of Bozen pages is shown. Notice that the text lines are well separated in the vertical direction, nonetheless, in some cases it is difficult to separate them on the horizontal direction.

The dataset contains 23.9 [6, 31] text lines per page on average, distributed over 4.6 [2, 9] layout regions on average. It is important to notice that each page can contain zero, one or more `heading` or `marginalia` layout regions to appear on any part of the document. Conversely, the `paragraph` regions always appear in the documents at least once.

Moreover, Bozen dataset can be obtained from the official link provided by its creators in https://doi.org/10.5281/zenodo.1297399.

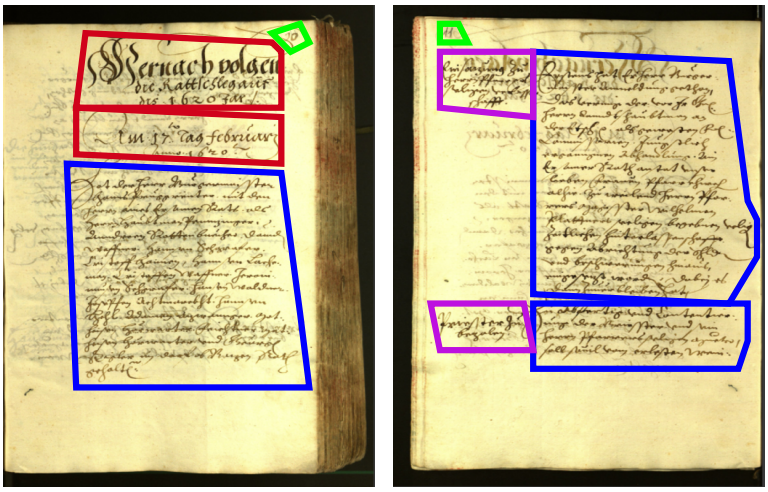


Figure B.3: Examples of pages with different layouts, belonging to the Bozen database. Blue: `paragraph`, red: `heading`, green: `page-number`, and violet: `marginalia`.

B.4 Competition on Baseline Detection in Archival Documents 2017 (cBAD-17)

This database was presented in [Die+17] for the ICDAR 2017 Competition on Baseline Detection in Archival Documents (cBAD). It is composed of 2035 annotated document page images that are collected from 9 different archives.

Two competition tracks and their corresponding partitions are defined on this corpus to test different characteristics of the submitted methods. Track A ("Simple Documents") is published with annotated text regions[3] and therefore aims to evaluate the quality of text line segmentation (216 pages for training and 539 for test).

The more challenging Track B ("Complex Documents") provides only the page area (270 pages for training and 1010 for test). Hence, baseline detection algorithms need to correctly locate text lines in the presence of marginalia, tables, and noise. The database comprises images with additional PAGE XMLs, which contain text regions and baseline annotations. In this dissertation we will focus on this part of the dataset as, as stated before, it is more complex.

Because it is composed of documents from 9 different collections, it is very heterogeneous, with documents ranging from simple pages with few or no text lines

[3]Only the polygon of the region is available, the class is not provided.

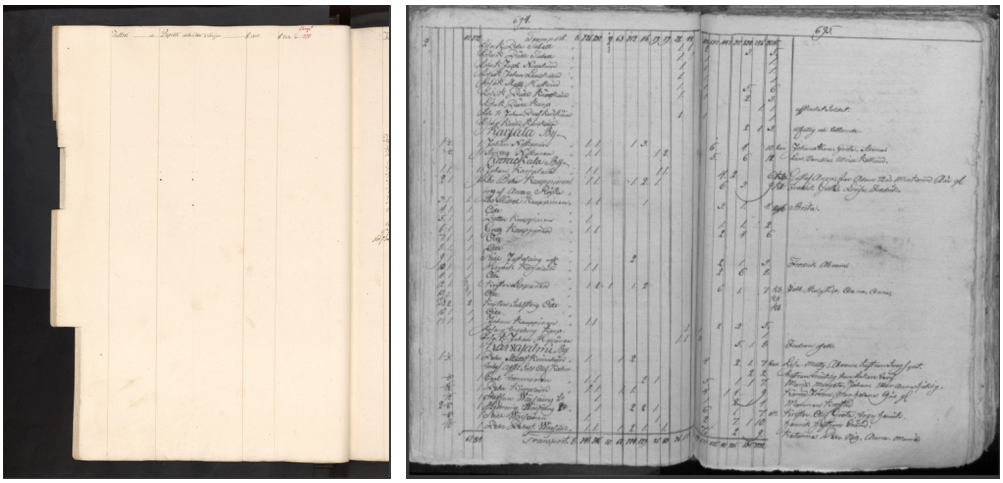


Figure B.4: Examples of pages with different number of text lines, belonging to the cBAD-17 dataset. In the left page only 6 text lines are defined, conversely the right page contains 370 text lines.

to pages with hundreds of text lines agglutinated on it. For instance, in Figure B.4 two examples are shown, the one in the left containing only 6 text lines whereas the page in the right contains 370 text lines. In general the dataset contains 80.3 [0, 472] text lines per page on average, nonetheless, they are distributed over a huge range (from 0 to 472).

This amalgam of layout styles and the variability in the density of the baselines in the documents cause the dataset to be very difficult to be processed by any DLA system. Therefore, therein lies the interest to be used in the Baseline Detection task.

Finally, the dataset can be obtained from the official link provided by its creators in http://doi.org/10.5281/zenodo.1208366.

B.5 Competition on Baseline Detection in Archival Documents 2019 (cBAD-19)

This database was presented in [Die+19] for the cBAD: ICDAR2019 Competition on Baseline Detection as the successor of the cBAD-17 dataset.

The cBAD-19 dataset is composed of 3021 document images, randomly divided into 775 images for training, 775 for validation and 1511 for test.

Following the same line as cBAD-17, cBAD-19 is a very heterogeneous dataset composed of document from different collections, hence different layouts are present on it. Moreover, the dataset is composed of a combination of handwritten documents, printed documents and documents that convey both styles. Also, conversely to cBAD-17, this dataset contains documents where the main source of information is not the text on it. For instance, drawings and photographs with captions, maps with labels, musical documents (where only the text is labeled), among others.

In Figure B.5 we show an example of documents of the cBAD-19. It is important to notice the variety of layouts and styles, as well as the variability on the number and distribution of text lines on each document. In particular, the datasets contains 63.87 [0, 2452] text lines per page on average. In effect, the number of text lines ranges from 0 to 2452, which makes the Baseline Detection problem very hard to address on it. Nonetheless, it is also what makes it interesting.

Moreover, the size of the images range from 1×10^6 px to 87×10^6 px, which can make this dataset very challenging to handle from the computational resources point of view[4].

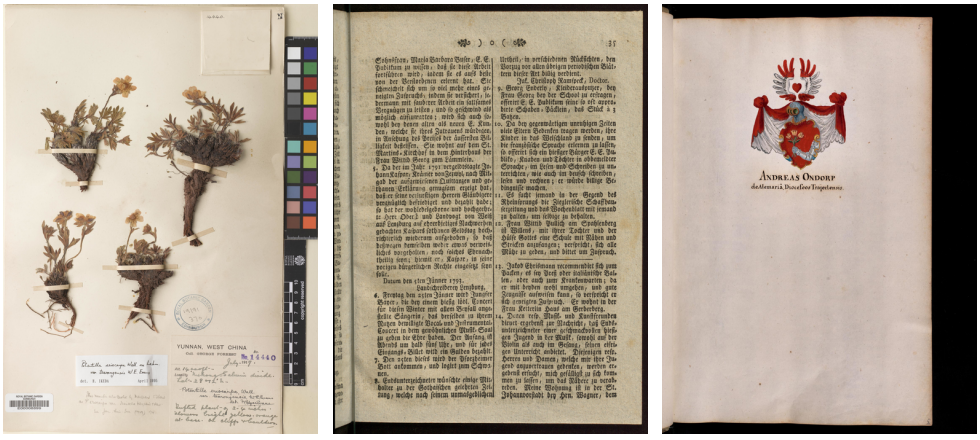


Figure B.5: Examples of pages with different number of text lines and layouts, belonging to the cBAD-19 dataset. In the left page only few text lines are defined, along with some kind of plants. In the middle case the page contains thousands of printed text lines arranges in two columns. Finally, in the case at the right, only two text lines can be observed.

[4]Normally 1×10^6 px are enough to handle handwritten documents, however, cBAD-19 contains maps and other large format documents.

The dataset is available online in the official link provided by its creators in http://zenodo.org/record/3234502.

B.6 Filand Renovated District Court Records (FCR)

The FCR dataset is a selection of 500 pages from the Renovated District Court Records (19th century), a large collection of the National Archives of Finland. The documents consist of records of deeds, mortgages, traditional life-annuity, among others.

This dataset contains images with one or two document pages, [5] annotated with text lines (baselines) and six different region types, namely: *page-number*, *marginalia*, *paragraph*, *paragraph2*, *table* and *table2*.

The blend of single and double-page images is a common complexity added to the DLA problem, and its subproblems like the reading order. Nonetheless, we select this dataset not only because of that complexity but also by the number of different regions and reading order distribution. In Figure B.6 we show an example of the pages and layout of the FCR dataset.

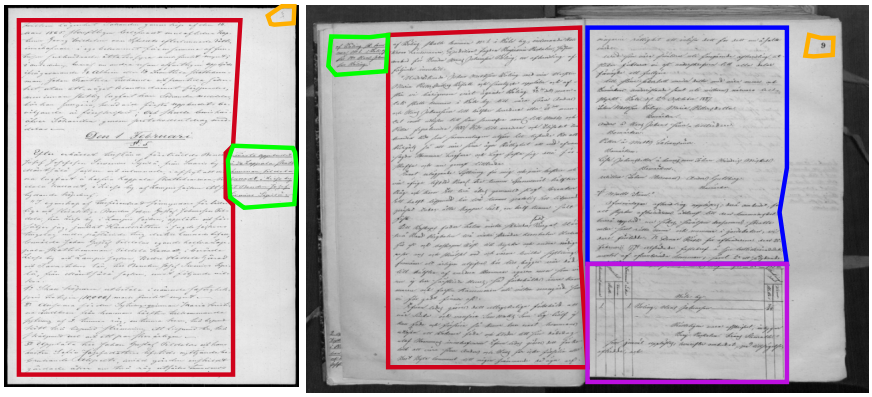


Figure B.6: Examples of pages with different layouts, belonging to the FCR database. Blue: `paragraph2`, red: `paragraph`, green: `marginalia`, violet: `table2`, and orange: `page-nuber`.

This dataset contains 63.7 [22, 131] text lines per page on average, distributed over 3.3 [1, 8] layout regions on average. Also, around 50% of the images are double-page.

The dataset is randomly divided into training, validation and test sets, 125 pages, 125 pages and 250 pages respectively. Moreover, the layout main characteristics are summarized in Table B.4.

[5] The word "page" is used here indistinctly to refer to a single or double-page image.

Table B.4: Main characteristics of the FCR dataset.

Region	#Regions				#Lines			
Name	Train	Val	Test	Total	Train	Val	Test	Total
page-number	86	87	168	341	92	87	169	348
marginalia	132	142	308	582	637	729	1659	3025
paragraph	183	195	381	714	4731	4707	9326	18764
paragraph2	59	70	129	258	2216	2525	4740	9481
table	2	5	12	19	55	95	261	411
table2	1	2	4	7	9	52	87	148

Furthermore, the dataset can be obtained online on the following link: https://zenodo.org/record/3945088

B.7 READ ABP Table (ABP)

The ABP dataset is a subset of the ABP_S_1847-1878 dataset, which contains information about the parishioners who died within the geographic boundaries of the various parishes of the Diocese of Passau between the years 1847 and 1878. This dataset contains a very heterogeneous set of pages where the main element is a table. It is composed of 111 manually annotated pages which amount to 29 752 text lines and 15 231 cells (i.e., 268.1 [21, 583] text lines per page on average). Only two different regions are defined, namely: *textRegion* for normal text regions and *tableRegion* for tables, distributed as 7.7 [1, 30] layout regions per page on average. In Figure B.7 we show an example of the type of documents layout on this dataset.

Figure B.7: Examples of pages with different layouts, belonging to the ABP database. Blue: `textRegion`, red: `tableRegion`.

The dataset is randomly divided into training, validation and test, 28 pages, 28 pages and 55 pages respectively. Moreover, the layout main characteristics are summarized in Table B.5.

Table B.5: Main characteristics of the ABP dataset.

Region	#Regions				#Lines			
Name	Train	Val	Test	Total	Train	Val	Test	Total
textRegion	190	182	370	742	213	205	424	842
tableRegion	28	28	58	114	6956	7636	14318	28910

Finally, the dataset can be downloaded from https://zenodo.org/record/1243098.

C *Extended results*

In this appendix we provide an extension of the results presented on Chapter 7, as it will be overcrowded to allocate all the information in that chapter. We encourage the interested reader to examine the results presented here as they compliment those previously presented. However, as the main discussion of the results is already addressed in Chapter 7 we provide no further discussion here.

C.1 Baseline Detection Extended Results

Table C.1: Baseline Detection results obtained using the proposed methods and different training strategies. Complete set of metrics. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	pre-trained	$P(\%)[CI]$	$R(\%)[CI]$	$F1(\%)[CI]$
		OHG		
Map-based	—	97.96 [97.70, 98.21]	98.13 [97.95, 98.30]	98.04 [97.82, 98.25]
Direct	—	97.08 [96.62, 97.50]	93.15 [92.08, 94.10]	95.07 [94.30, 95.77]
Map-based	PubLayNet	98.29 [98.09, 98.49]	98.30 [98.14, 98.45]	98.29 [98.11, 98.47]
Direct	PubLayNet	97.83 [97.50, 98.12]	96.23 [95.52, 96.84]	97.02 [96.50, 97.48]
Direct	ImageNet	98.00 [97.68, 98.29]	96.46 [95.79, 97.04]	97.22 [96.73, 97.66]
		VORAU-253		
Map-based	—	94.04 [92.46, 95.53]	98.50 [97.88, 99.05]	96.22 [95.09, 97.26]
Direct	—	95.18 [93.60, 96.56]	97.75 [97.02, 98.42]	96.45 [95.28, 97.48]
Map-based	PubLayNet	92.63 [90.79, 94.33]	98.22 [97.57, 98.82]	95.34 [94.06, 96.52]
Direct	PubLayNet	95.33 [93.81, 96.71]	98.24 [97.54, 98.85]	96.76 [95.64, 97.77]
Direct	ImageNet	98.03 [97.11, 98.83]	97.79 [97.03, 98.48]	97.91 [97.07, 98.65]

C.1.1 Effect of Training Data Size Results

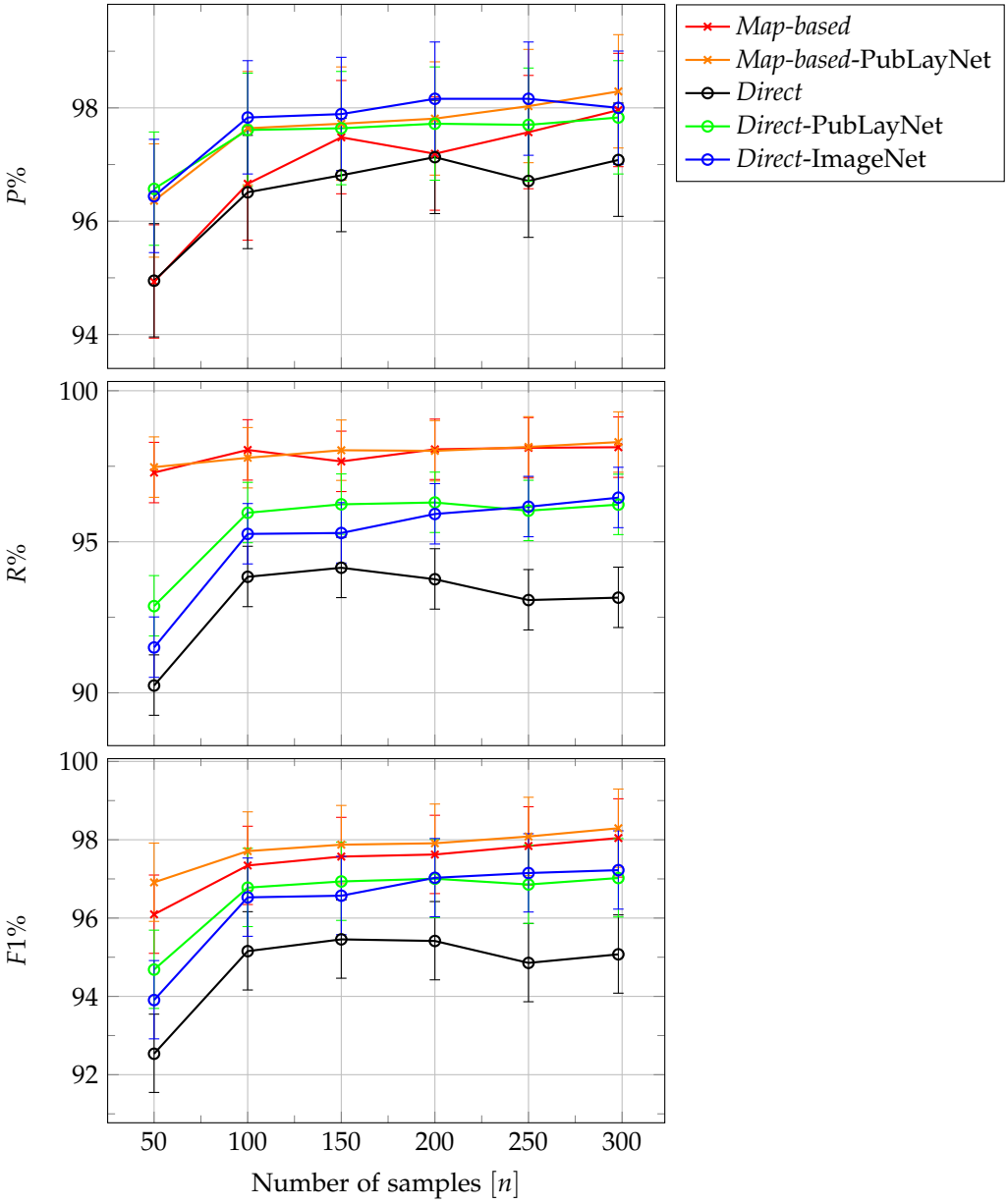


Figure C.1: Effect of training data size on Baseline Detection metrics for OHG dataset.

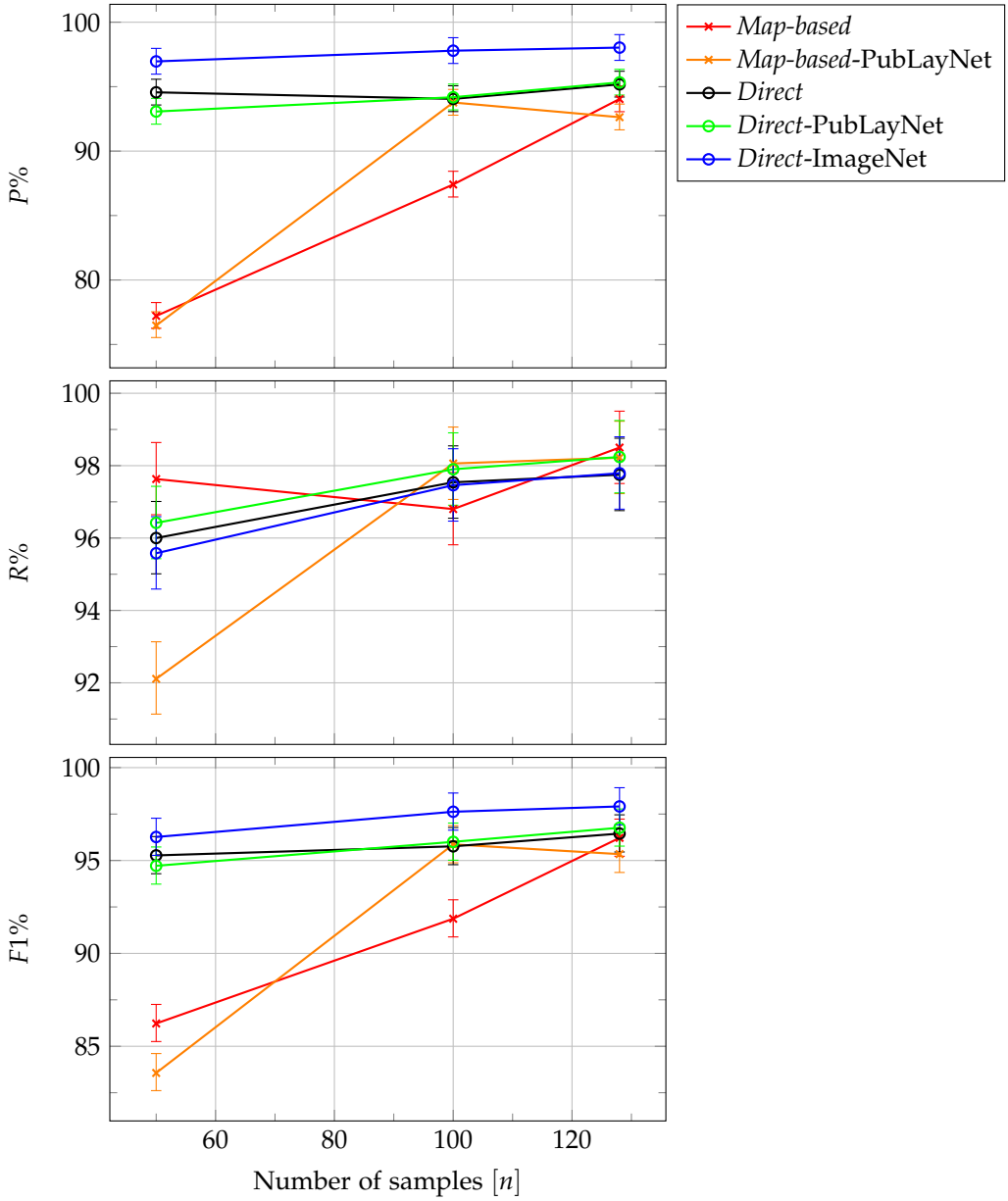


Figure C.2: Effect of training data size on Baseline Detection metrics for VORAU-253 dataset.

C.2 Region Segmentation Extended Results

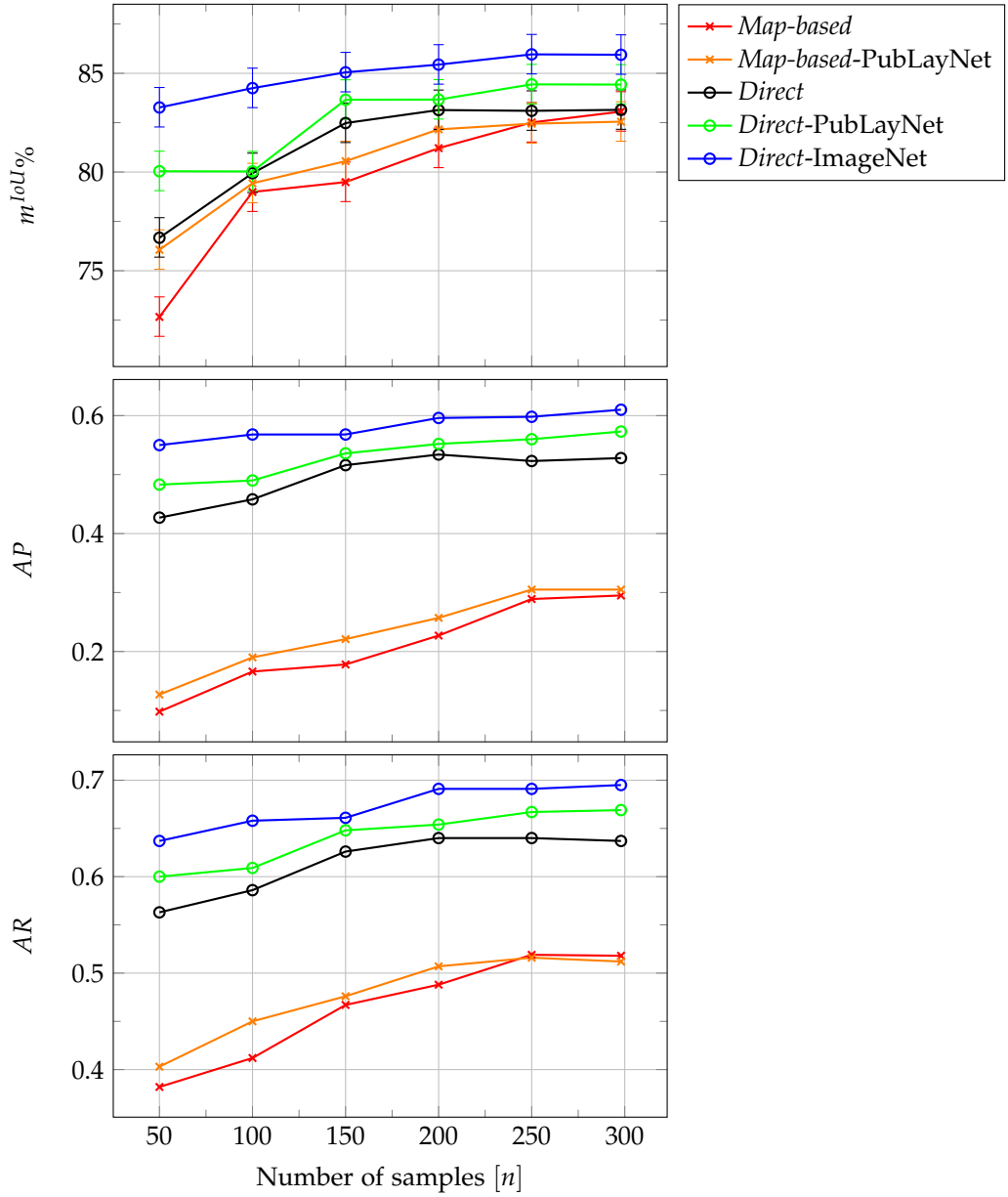


Figure C.3: Effect of training data size on Region Segmentation metrics for OHG dataset.

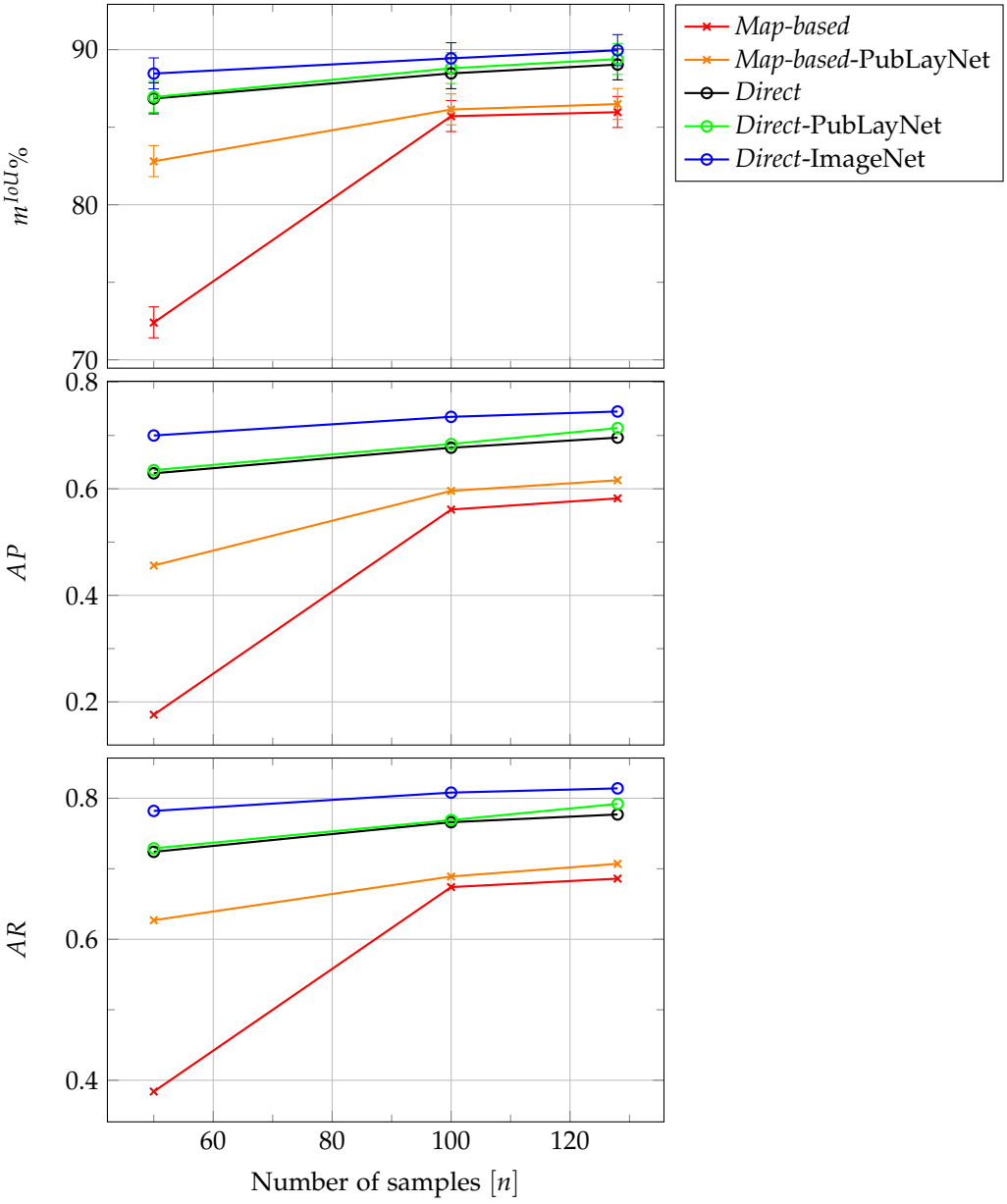


Figure C.4: Effect of training data size on Region Segmentation metrics for VORAU-253 dataset.

Table C.2: Region Segmentation results obtained using the proposed methods and different training strategies (using the complete training set). Complete set of metrics. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	pre-trained	pixel acc. (%)[CI]	mean acc. (%)[CI]	m^{IoU} (%)[CI]	$f.w^{IoU}$ (%)[CI]	AP	AP_{50}	AP_{75}	AR
				OHG					
Map-based	—	83.06 [82.16, 83.91]	83.06 [82.16, 83.91]	83.06 [82.16, 83.91]	83.06 [82.16, 83.91]	0.30	0.59	0.25	0.52
Direct	—	83.15 [81.95, 84.28]	83.15 [81.95, 84.28]	83.15 [81.95, 84.28]	83.15 [81.95, 84.28]	0.53	0.77	0.53	0.64
Map-based	PubLayNet	82.55 [81.70, 83.39]	82.55 [81.70, 83.39]	82.55 [81.70, 83.39]	82.55 [81.70, 83.39]	0.31	0.60	0.25	0.51
Direct	PubLayNet	84.43 [83.32, 85.49]	84.43 [83.32, 85.49]	84.43 [83.32, 85.49]	84.43 [83.32, 85.49]	0.57	0.84	0.58	0.67
Direct	ImageNet	85.94 [84.98, 86.80]	85.94 [84.98, 86.80]	85.94 [84.98, 86.80]	85.94 [84.98, 86.80]	0.61	0.90	0.64	0.70
				VORAU-253					
Map-based	—	85.97 [84.60, 87.05]	85.97 [84.60, 87.05]	85.97 [84.60, 87.05]	85.97 [84.60, 87.05]	0.58	0.86	0.71	0.69
Direct	—	89.04 [87.79, 89.85]	89.04 [87.79, 89.85]	89.04 [87.79, 89.85]	89.04 [87.79, 89.85]	0.70	0.96	0.86	0.78
Map-based	PubLayNet	86.49 [85.42, 87.21]	86.49 [85.42, 87.21]	86.49 [85.42, 87.21]	86.49 [85.42, 87.21]	0.62	0.90	0.76	0.71
Direct	PubLayNet	89.39 [88.16, 90.21]	89.39 [88.16, 90.21]	89.39 [88.16, 90.21]	89.39 [88.16, 90.21]	0.71	0.96	0.89	0.79
Direct	ImageNet	89.96 [88.74, 90.75]	89.96 [88.74, 90.75]	89.96 [88.74, 90.75]	89.96 [88.74, 90.75]	0.75	0.97	0.91	0.81

C.3 Integrated Results

Table C.3: Results obtained using the integrated approach for the Baseline Detection task. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	pre-trained	$P(\%)[CI]$	$R(\%)[CI]$	$F1(\%)[CI]$
		OHG		
Map-based	—	94.68 [93.62, 95.66]	97.90 [97.68, 98.10]	96.26 [95.61, 96.86]
Direct	—	96.99 [96.52, 97.41]	91.74 [90.55, 92.82]	94.29 [93.44, 95.06]
Map-based	PubLayNet	94.37 [93.40, 95.26]	97.86 [97.62, 98.08]	96.08 [95.46, 96.65]
Direct	PubLayNet	97.99 [97.68, 98.27]	96.20 [95.54, 96.77]	97.09 [96.60, 97.51]
Direct	ImageNet	98.01 [97.70, 98.30]	95.34 [94.63, 95.96]	96.66 [96.14, 97.12]
		VORAU-253		
Map-based	—	95.19 [93.62, 96.57]	95.19 [93.62, 96.57]	96.78 [95.65, 97.76]
Direct	—	95.71 [94.08, 97.13]	95.71 [94.08, 97.13]	95.53 [94.19, 96.72]
Map-based	PubLayNet	94.60 [93.37, 95.78]	94.60 [93.37, 95.78]	96.30 [95.33, 97.20]
Direct	PubLayNet	95.81 [94.34, 97.11]	95.81 [94.34, 97.11]	95.94 [94.66, 97.09]
Direct	ImageNet	98.03 [97.11, 98.82]	98.03 [97.11, 98.82]	96.01 [94.84, 97.07]

Table C.4: Results obtained using the integrated approach for the Region Segmentation task. Nonparametric Bootstrapping confidence intervals (CI) at 95%, using 10000 repetitions.

Method	pre-trained	pixel acc. (%)[CI]	mean acc. (%)[CI]	m^{IoU} (%)[CI]	$f.w^{IoU}$ (%)[CI]	AP	AP_{50}	AP_{75}	AR
				OHG					
Map-based	—	80.54 [79.51, 81.54]	80.54 [79.51, 81.54]	80.54 [79.51, 81.54]	80.54 [79.51, 81.54]	0.31	0.59	0.27	0.48
Direct	—	81.36 [80.05, 82.62]	81.36 [80.05, 82.62]	81.36 [80.05, 82.62]	81.36 [80.05, 82.62]	0.51	0.75	0.52	0.61
Map-based	PubLayNet	81.61 [80.66, 82.53]	81.61 [80.66, 82.53]	81.61 [80.66, 82.53]	81.61 [80.66, 82.53]	0.33	0.66	0.28	0.49
Direct	PubLayNet	83.40 [82.20, 84.55]	83.40 [82.20, 84.55]	83.40 [82.20, 84.55]	83.40 [82.20, 84.55]	0.55	0.80	0.58	0.65
Direct	ImageNet	85.46 [84.49, 86.37]	85.46 [84.49, 86.37]	85.46 [84.49, 86.37]	85.46 [84.49, 86.37]	0.60	0.84	0.63	0.68
				VORAU-253					
Map-based	—	86.72 [85.63, 87.45]	86.72 [85.63, 87.45]	86.72 [85.63, 87.45]	86.72 [85.63, 87.45]	0.61	0.89	0.75	0.70
Direct	—	88.80 [87.54, 89.68]	88.80 [87.54, 89.68]	88.80 [87.54, 89.68]	88.80 [87.54, 89.68]	0.69	0.96	0.86	0.77
Map-based	PubLayNet	86.65 [85.53, 87.47]	86.65 [85.53, 87.47]	86.65 [85.53, 87.47]	86.65 [85.53, 87.47]	0.61	0.89	0.75	0.70
Direct	PubLayNet	89.69 [88.49, 90.48]	89.69 [88.49, 90.48]	89.69 [88.49, 90.48]	89.69 [88.49, 90.48]	0.72	0.97	0.89	0.79
Direct	ImageNet	89.99 [88.79, 90.75]	89.99 [88.79, 90.75]	89.99 [88.79, 90.75]	89.99 [88.79, 90.75]	0.75	0.97	0.91	0.81

C.4 Reading Order Results

Table C.5: Layout element ordering results for different metrics, tasks and decoders. Reported figures are page or region averages of values of $\rho(t, \nu)$ (in %) and $K(t, \nu)$ (absolute numbers of swaps). Tasks correspond to: ordering Lines at Page level (LP), Regions at Page level (RP), Lines at Region level (LR) and Lines obtained through Hierarchical processing via RP, but evaluated at Page level (LHP). The decoders are: Top Bottom Left Right (TBLR), Greedy, First Decide Then Decode (FDTD) and Brute Force (BF). Order relation probabilities are learned using a Multilayer Perceptron. Each result is the average over 10 randomly initialized experiments. In both metrics the lower the better.

Metric	$\rho(\boldsymbol{t}, \boldsymbol{\nu})$ (%)				$K(\boldsymbol{t}, \boldsymbol{\nu})$			
Decoder	LP	RP	LR	LHP	LP	RP	LR	LHP
	OHG							
TBLR	2.875	9.348	0.061	3.511	12.899	0.614	0.013	0.671
Greedy	0.504	0.125	0.053	0.035	2.517	0.009	0.012	0.065
FDTD	0.498	0.134	0.053	0.035	2.447	0.012	0.012	0.069
	FCR							
TBLR	31.838	28.431	0.654	31.379	606.976	1.860	1.859	8.324
Greedy	1.063	1.629	0.336	1.074	18.372	0.113	1.157	4.136
FDTD	0.699	1.692	0.338	1.152	11.544	0.128	1.153	4.138
	ABP							
TBLR	10.989	16.900	0.909	8.785	4020.111	5.148	221.111	1733.092
Greedy	4.707	6.012	0.605	4.778	2218.82	0.963	114.199	893.404
FDTD	4.679	5.920	0.603	4.713	2211.32	0.953	113.842	890.606

List of Figures

List of Tables

Bibliography

[ADF10] Alexe, B., Deselaers, T., and Ferrari, V. "What is an object?" In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 73–80.

[Alv+13] Alvaro, F. et al. "Page Segmentation of Structured Documents Using 2D Stochastic Context-Free Grammars". In: *6th Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA), LNCS 7887*. 2013, pp. 133–140.

[ASK18] Ares Oliveira, S., Seguin, B., and Kaplan, F. "dhSegment: A Generic Deep-Learning Approach for Document Segmentation". In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2018, pp. 7–12.

[BB08] Bottou, L. and Bousquet, O. "The Tradeoffs of Large Scale Learning". In: *Advances in Neural Information Processing Systems 20 (NIPS 2007)*. Ed. by Platt, J. et al. NIPS Foundation, 2008, pp. 161–168.

[Ber+07] Bertolami, R. et al. "Non-Uniform Slant Correction for Handwritten Text Line Recognition". In: *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*. Vol. 1. 2007, pp. 18–22.

[BI11] Baechler, M. and Ingold, R. "Multi Resolution Layout Analysis of Medieval Manuscripts Using Dynamic MLP". In: *2011 International Conference on Document Analysis and Recognition*. Sept. 2011, pp. 1185–1189.

[Bis06] Bishop, C. *Pattern Recognition and Machine Learning*. Springer, Jan. 2006.

[Blu15] Bluche, T. "Deep Neural Networks for Large Vocabulary Handwritten Text Recognition". PhD thesis. Université Paris-Sud, May 2015.

[BN04] Bisani, M. and Ney, H. "Bootstrap estimates for confidence intervals in ASR performance evaluation". In: *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 1. 2004, pp. I–409.

[Bor+20] Boroş, E. et al. "A comparison of sequential and combined approaches for named entity recognition in a corpus of handwritten medieval charters". In: *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2020, pp. 79–84.

[Bos+18] Bosch Campos, V. et al. “Text Line Extraction Based on Distance Map Features and Dynamic Programming”. In: *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2018, pp. 357–362.

[Bos20] Bosch Campos, V. “Advances in Document Layout Analysis”. PhD thesis. Universitat Politècnica de València, Jan. 2020.

[Bre03] Breuel, T. M. “High Performance Document Layout Analysis”. In: *2003 Symposium on Document Image Understanding (SDIUT'03)*. 2003.

[Bri90] Bridle, J. S. “Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition”. In: *Neurocomputing*. Springer, 1990, pp. 227–236.

[BTV12] Bosch, V., Toselli, A. H., and Vidal, E. “Statistical Text Line Analysis in Handwritten Documents”. In: *2012 International Conference on Frontiers in Handwriting Recognition*. 2012, pp. 201–206.

[Car93] Caruana, R. “Multitask Learning: A Knowledge-Based Source of Inductive Bias”. In: *Proceedings of the Tenth International Conference on Machine Learning*. Morgan Kaufmann, 1993, pp. 41–48.

[Cat+98] Cattoni, R. et al. *Geometric layout analysis techniques for document image understanding: a review*. Tech. rep. ITC-irst, 1998.

[CL96] Casey, R. G. and Lecolinet, E. “A survey of methods and strategies in character segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 18, no. 7 (1996), pp. 690–706.

[Coh+13] Cohen, R. et al. “Robust text and drawing segmentation algorithm for historical documents”. In: *ACM International Conference Proceeding Series*. Aug. 2013, pp. 110–117.

[CW09] Chen, Y.-L. and Wu, B.-F. “A multi-plane approach for text segmentation of complex document images”. In: *Pattern Recognition* vol. 42, no. 7 (2009), pp. 1419–1444.

[Cyb89] Cybenko, G. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* vol. 2, no. 4 (1989), pp. 303–314.

[DC12] Das Gupta, J. and Chanda, B. “Novel methods for slope and slant correction of off-line handwritten text word”. In: *2012 Third International Conference on Emerging Applications of Information Technology*. 2012, pp. 295–298.

[Den+09] Deng, J. et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[DHS01] Duda, R. O., Hart, P. E., and Stork, D. G. *Pattern Classification*. 2nd ed. New York: Wiley, 2001.

[Die+17] Diem, M. et al. "cBAD: ICDAR2017 Competition on Baseline Detection". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. Nov. 2017, pp. 1355–1360.

[Die+19] Diem, M. et al. "cBAD: ICDAR2019 Competition on Baseline Detection". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, pp. 1494–1498.

[DP90] Davey, B. A. and Priestley, H. A. *Introduction to lattices and order*. Cambridge: Cambridge University Press, 1990.

[DV18] Dumoulin, V. and Visin, F. *A guide to convolution arithmetic for deep learning*. 2018. arXiv: 1603.07285 [stat.ML].

[Efr87] Efron, B. "Better Bootstrap Confidence Intervals". In: *Journal of the American Statistical Association* vol. 82, no. 397 (1987), pp. 171–185.

[EGO17] Eskenazi, S., Gomez-Krämer, P., and Ogier, J.-M. "A comprehensive survey of mostly textual document segmentation algorithms since 2008". In: *Pattern Recognition* vol. 64 (2017), pp. 1–14.

[Erh+14] Erhan, D. et al. "Scalable object detection using deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 2147–2154.

[Fin+18] Fink, M. et al. "Baseline Detection in Historical Documents Using Convolutional U-Nets". In: *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. 2018, pp. 37–42.

[FT12] Fernández, F. C. and Terrades, O. R. "Document segmentation using Relative Location Features". In: *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. Nov. 2012, pp. 1562–1565.

[Gao+19] Gao, L. et al. "ICDAR 2019 Competition on Table Detection and Recognition (cTDaR)". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, pp. 1510–1515.

[GBC16] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.

[GFG06] Graves, A., Fernández, S., and Gomez, F. "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks". In: *In Proceedings of the International Conference on Machine Learning, ICML 2006*. 2006, pp. 369–376.

[Gio+17] Giotis, A. P. et al. "A survey of document image word spotting techniques". In: *Pattern Recognition* vol. 68 (2017), pp. 310–332.

[Góm10] Gómez, V. R. "Multimodal Interactive Transcription of Handwritten Text Images". PhD thesis. Universitat Politècnica de València, Sept. 2010.

[GPC97] Gatos, B., Papamarkos, N., and Chamzas, C. "Skew detection and text line position determination in digitized documents". In: *Pattern Recognition* vol. 30, no. 9 (1997), pp. 1505–1519.

[Gra+16] Grana, C. et al. "YACCLAB - Yet Another Connected Components Labeling Benchmark". In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. (Cancun, Mexico). Springer, Dec. 2016, pp. 3109–3114.

[Gra+21] Granell, E. et al. "Reducing the Human Effort in Text Line Segmentation for Historical Documents". In: *Document Analysis and Recognition – ICDAR 2021*. Cham: Springer International Publishing, 2021, pp. 523–537.

[Grü+17] Grüning, T. et al. "READ-BAD: A New Dataset and Evaluation Scheme for Baseline Detection in Archival Documents". In: *CoRR* vol. abs/1705.03311 (2017). arXiv: 1705.03311.

[Grü+19] Grüning, T. et al. "A two-stage method for text line detection in historical documents". In: *International Journal of Document Analysis and Recognition (IJDAR)* vol. 22, no. 3 (2019), pp. 285–302.

[GW08] Gonzalez, R. C. and Woods, R. E. *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall, 2008.

[He+16] He, K. et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[He+17] He, K. et al. "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2980–2988.

[Hos+16] Hosang, J. et al. "What Makes for Effective Detection Proposals?" In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 38, no. 4 (2016), pp. 814–830.

[Hun75] Hunt, E. B. *Artificial Intelligence*. New York, USA: Academic Press, 1975.

[Iso+17] Isola, P. et al. "Image-to-Image Translation with Conditional Adversarial Networks". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5967–5976.

[Jar+09] Jarrett, K. et al. "What is the best multi-stage architecture for object recognition?" In: *2009 IEEE 12th International Conference on Computer Vision*. 2009, pp. 2146–2153.

[JET19] Jaume, G., Ekenel, H. K., and Thiran, J. "FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents". In: *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*. Vol. 2. Los Alamitos, CA, USA: IEEE Computer Society, Sept. 2019, pp. 1–6.

[KB15] Kingma, D. P. and Ba, J. "Adam: A method for stochastic optimization". In: *3rd International Conference on Learning Representations (ICLR)* (2015).

[Ken38] Kendall, M. G. "A New Measure of Rank Correlation". In: *Biometrika* vol. 30, no. 1/2 (1938), pp. 81–93.

[KG98] Kim, G. and Govindaraju, V. "Handwritten phrase recognition as applied to street name images". In: *Pattern Recognition* vol. 31, no. 1 (1998), pp. 41–51.

[KGC17] Kukačka, J., Golkov, V., and Cremers, D. *Regularization for Deep Learning: A Taxonomy*. 2017. arXiv: 1710.10686 [cs.LG].

[Kol56] Kolmogorov, A. N. *Fundations of the Theory of Probability*. Chelsea Publishing Company, New York, 1956.

[KV10] Kumar, R. and Vassilvitskii, S. "Generalized distances between rankings". In: Jan. 2010, pp. 571–580.

[LCC08] Lemaitre, A., Camillerapp, J., and Coüasnon, B. "Multiresolution cooperation makes easier document structure recognition". In: *International Journal of Document Analysis and Recognition (IJDAR)* vol. 11, no. 2 (Nov. 2008), pp. 97–109.

[LeC+90] LeCun, Y. et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems*. Ed. by Touretzky, D. Vol. 2. Morgan-Kaufmann, 1990.

[LeC89] LeCun, Y. *Generalization and network design strategies*. Tech. rep. CRG-TR-89-4. University of Toronto, 1989.

[Lee+02] Lee, J.-Y. et al. "Automatic generation of structured hyperdocuments from document images". In: *Pattern Recognition* vol. 35, no. 2 (2002), pp. 485–503.

[Lin+15] Lin, T.-Y. et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV].

[Lin+17] Lin, T.-Y. et al. "Feature Pyramid Networks for Object Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[Lou+09] Louloudis, G. et al. “Text line and word segmentation of handwritten documents”. In: *Pattern Recognition* vol. 42, no. 12 (2009). New Frontiers in Handwriting Recognition, pp. 3169–3183.

[LSD15] Long, J., Shelhamer, E., and Darrell, T. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[LZT07] Likforman-Sulem, L., Zahour, A., and Taconet, B. “Text line segmentation of historical documents: a survey”. In: *International Journal of Document Analysis and Recognition (IJDAR)* vol. 9 (2007), pp. 123–138.

[MCB08] Malerba, D., Ceci, M., and Berardi, M. “Machine Learning for Reading Order Detection in Document Image Understanding”. In: *Machine Learning in Document Analysis and Recognition*. Springer Berlin Heidelberg, 2008, pp. 45–69.

[MH16] Ma, X. and Hovy, E. “End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1064–1074.

[MHN13] Maas, A. L., Hannun, A. Y., and Ng, A. Y. “Rectifier nonlinearities improve neural network acoustic models”. In: *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. 2013.

[Mit97] Mitchell, T. M. *Machine Learning*. McGraw-Hill, Inc., USA., 1997.

[MLF14] Mota, D. F., Lladós, J., and Fornés, A. “A graph-based approach for segmenting touching lines in historical handwritten documents”. In: *Int. J. Document Anal. Recognit.* Vol. 17, no. 3 (2014), pp. 293–312.

[MN95] Mahadevan, U. and Nagabushnam, R. C. “Gap metrics for word separation in handwritten lines”. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. 1995, 124–127 vol.1.

[Moy+15] Moysset, B. et al. “Paragraph text segmentation into lines with Recurrent Neural Networks”. In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 2015, pp. 456–460.

[MP43] McCulloch, W. S. and Pitts, W. “A logical calculus of the ideas immanent in nervous activity”. In: *The bulletin of mathematical biophysics* vol. 5, no. 4 (1943), pp. 115–133.

[MV93] Marzal, A. and Vidal, E. “Computation of normalized edit distance and applications”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 15, no. 9 (1993), pp. 926–932.

[Nag00] Nagy, G. "Twenty Years of Document Image Analysis in PAMI". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* vol. 22 (Feb. 2000), pp. 38–62.

[NJ02] Ng, A. and Jordan, M. "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes". In: *Advances in Neural Information Processing Systems*. Ed. by Dietterich, T., Becker, S., and Ghahramani, Z. Vol. 14. MIT Press, 2002.

[NNC19] Naoum, A., Nothman, J., and Curran, J. "Article Segmentation in Digitised Newspapers with a 2D Markov Model". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, pp. 1007–1014.

[Pap+02] Papineni, K. et al. "BLEU: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL '02. Philadelphia, Pennsylvania, 2002, pp. 311–318.

[PDM19] Prasad, A., Déjean, H., and Meunier, J. "Versatile Layout Understanding via Conjugate Graph". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. 2019, pp. 287–294.

[Pil01] Pilu, M. "Undoing page curl distortion using applicable surfaces". In: *Proceedings 2001 International Conference on Image Processing*. Vol. 1. 2001, 237–240 vol.1.

[Pri+20] Prieto, J. R. et al. "Text Content Based Layout Analysis". In: *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2020, pp. 258–263.

[Pru+19] Prusty, A. et al. "Indiscapes: Instance segmentation networks for layout parsing of historical indic manuscripts". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 999–1006.

[Pui18] Puigcerver, J. "A Probabilistic Formulation of Keyword Spotting". PhD thesis. Universitat Politècnica de València, Nov. 2018.

[PV94] Perez, J.-C. and Vidal, E. "Optimum polygonal approximation of digitalized curves". In: *Pattern Recognition Letters* (1994).

[QTV19] Quirós, L., Toselli, A. H., and Vidal, E. "Multi-task Layout Analysis of Handwritten Musical Scores". In: *Iberian Conference on Pattern Recognition and Image Analysis*. Springer. 2019, pp. 123–134.

[Qui+17] Quirós, L. et al. "Interactive Layout Detection". In: *8th Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*. Cham: Springer International Publishing, 2017, pp. 161–168.

[Qui+18a] Quirós, L. et al. "From HMMs to RNNs: Computer-assisted Transcription of a Handwritten Notarial Records Collection". In: *International Conference on Frontiers in Handwriting Recognition (ICFHR)*. Aug. 2018, pp. 116–121.

[Qui+18b] Quirós, L. et al. *Oficio de Hipotecas de Girona. A dataset of Spanish notarial deeds (18th Century) for Handwritten Text Recognition and Layout Analysis of historical documents.* https://doi.org/10.5281/zenodo.1322666. July 2018.

[Qui18] Quirós, L. "Multi-Task Handwritten Document Layout Analysis". In: *ArXiv e-prints, 1806.08852* (2018). arXiv: 1806.08852 [cs.CV].

[QV21] Quirós, L. and Vidal, E. "Learning to Sort Handwritten Text Lines in Reading Order through Estimated Binary Order Relations". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021, pp. 7661–7668.

[Ren+16] Ren, S. et al. "Faster R-CNN: towards real-time object detection with region proposal networks". In: *IEEE transactions on pattern analysis and machine intelligence* vol. 39, no. 6 (2016), pp. 1137–1149.

[Ren+18] Renton, G. et al. "Fully convolutional network with dilated convolutions for handwritten text line segmentation". In: *International Journal of Document Analysis and Recognition (IJDAR)* vol. 21, no. 3 (2018), pp. 177–186.

[RF17] Redmon, J. and Farhadi, A. "YOLO9000: Better, Faster, Stronger". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 6517–6525.

[RFB15] Ronneberger, O., Fischer, P., and Brox, T. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Navab, N. et al. Cham: Springer International Publishing, 2015, pp. 234–241.

[RHW86] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. "Learning representations by back-propagating errors". In: *nature* vol. 323, no. 6088 (1986), pp. 533–536.

[Rom+15] Romero, V. et al. "Influence of text line segmentation in Handwritten Text Recognition". In: *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*. 2015, pp. 536–540.

[Ros58] Rosenblatt, F. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* vol. 65, no. 6 (1958), p. 386.

[SA85] Suzuki, S. and Abe, K. “Topological structural analysis of digitized binary images by border following”. In: *Comput. Vis. Graph. Image Process.* Vol. 30 (1985), pp. 32–46.

[San+11] Sande, K. E. A. van de et al. “Segmentation as selective search for object recognition”. In: *2011 International Conference on Computer Vision*. 2011, pp. 1879–1886.

[Sán+19] Sánchez, J.-A. et al. “A set of benchmarks for Handwritten Text Recognition on historical documents”. In: *Pattern Recognition* vol. 94 (2019), pp. 122–134.

[SC94] Seni, G. and Cohen, E. “External word segmentation of off-line handwritten text lines”. In: *Pattern Recognition* vol. 27, no. 1 (1994), pp. 41–52.

[Sha96] Shapiro, L. G. “Connected Component Labeling and Adjacency Graph Construction”. In: *Topological Algorithms for Digital Image Processing*. Ed. by Kong, T. Y. and Rosenfeld, A. Vol. 19. Machine Intelligence and Pattern Recognition. North-Holland, 1996, pp. 1–30.

[Sim+16] Simistira, F. et al. “DIVA-HisDB: A Precisely Annotated Large Dataset of Challenging Medieval Manuscripts”. In: *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2016, pp. 471–476.

[Sim+17] Simistira, F. et al. “ICDAR2017 Competition on Layout Analysis for Challenging Medieval Manuscripts”. In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. 2017, pp. 1361–1370.

[SK19] Shorten, C. and Khoshgoftaar, T. M. “A survey on Image Data Augmentation for Deep Learning”. In: *Journal of Big Data* vol. 6, no. 1 (2019), pp. 60–108.

[SMJ19] Saha, R., Mondal, A., and Jawahar, C. “Graphical object detection in document images”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 51–58.

[SSP03] Simard, P. Y., Steinkraus, D., and Platt, J. “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”. In: Institute of Electrical and Electronics Engineers, Inc., Aug. 2003.

[Stu+19] Studer, L. et al. “A comprehensive study of imagenet pre-training for historical document image analysis”. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 720–725.

[Suz+85] Suzuki, S. et al. "Topological structural analysis of digitized binary images by border following". In: *Computer vision, graphics, and image processing* vol. 30, no. 1 (1985), pp. 32–46.

[SZ14] Simonyan, K. and Zisserman, A. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[Ten+17] Tensmeyer, C. et al. "PageNet: Page Boundary Extraction in Historical Handwritten Documents". In: *arXiv preprint arXiv:1709.01618* (2017).

[Tos+18] Toselli, A. et al. *HTR Dataset ICFHR 2016*. https://doi.org/10.5281/zenodo.1297399. Feb. 2018.

[TVC11] Toselli, A. H., Vidal, E., and Casacuberta, F. *Multimodal Interactive Pattern Recognition and Applications*. 1st. Springer Publishing Company, Incorporated, 2011.

[VB05] Varga, T. and Bunke, H. "Tree structure for word extraction from handwritten text lines". In: *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*. 2005, 352–356 Vol. 1.

[VJ01] Viola, P. and Jones, M. "Robust real-time face detection". In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 2. 2001, pp. 747–747.

[VT21] Vidal, E. and Toselli, A. H. "Probabilistic Indexing and Search for Hyphenated Words". In: *Document Analysis and Recognition – ICDAR 2021*. Cham: Springer International Publishing, 2021, pp. 426–442.

[Wei+13] Wei, H. et al. "Evaluation of SVM, MLP and GMM Classifiers for Layout Analysis of Historical Documents". In: *2013 12th International Conference on Document Analysis and Recognition*. Aug. 2013, pp. 1220–1224.

[Wol94] Wolberg, G. *Digital Image Warping*. 1st. Washington, DC, USA: IEEE Computer Society Press, 1994.

[ZC15] Zhong, G. and Cheriet, M. "Tensor representation learning based image patch analysis for text identification and recognition". In: *Pattern Recognition* vol. 48, no. 4 (2015), pp. 1211–1224.

[ZT03] Zhang, Z. and Tan, C. L. "Correcting document image warping based on regression of curved text lines". In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* 2003, 589–593 vol.1.

[ZTY19] Zhong, X., Tang, J., and Yepes, A. J. "Publaynet: largest dataset ever for document layout analysis". In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2019, pp. 1015–1022.