UNIVERSITAT POLITÈCNICA DE VALÈNCIA
DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

PATTERN RECOGNITION AND
HUMAN LANGUAGE TECHNOLOGY GROUP

IARFID Master Thesis

Handwritten Text Line Detection and Classification based on HMMs

Author: Vicente Bosch Campos

Advisor: Enrique Vidal Ruiz
Co-advisor: Alejandro Héctor Toselli

June 28, 2012

ACKNOWLEDGEMENTS

Many people have helped me out in order to let me conclude this research. Thanks to my parents for teaching me the value of hard work, to my friends in the PRHLT lab for making the journey much more enjoyable and finally to Enrique Vidal and Alejandro Héctor Toselli for their time, patience and knowledge without which this work could not have been completed.

The research leading to these results has received funding from the MIPRCV "Consolider Ingenio 2010" program (CSD2007-00018), MITTRAL (TIN2009-14633-C03-01) and also Univ. Politécnica Valencia (PAID-05-11).

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Chapter Outline

1.1 Introduction

Pattern recognition has as one of its major objectives to be able to reproduce human level perception capabilities in a system. Inside this major goal of perception we can consider the task of reading text, either handwritten or printed.

Document Layout Analysis (DLA) is the process by which regions of interest in a document image are detected and categorized. This makes it an important and necessary task in any text recognition/transcription related tasks. Within DLA we find a *page segmentation* phase, an important initial step in charge of dividing the document image into homogeneous zones.

Text line analysis and detection (TLAD) is another DLA task embedded inside *page segmentation* that constitutes an essential step in any modern text recognition and transcription systems that require input in the form of text line images. For example, text/image alignment [9], fully automatic handwritten text recognition [1, 7, 10] (HTR) or computer assisted transcription of text images (CATTI), where the users participate interactively in the actual transcription process [8]. Furthermore, due to the dependence ,such systems have, on input quality, TLAD often has a significant impact on the final accuracy.

Detection of handwritten text lines in an image entails a greater difficulty, in comparison with printed text lines, due to the inherent properties of handwritten text: variable inter-line spacing, overlapping and touching strokes of adjacent handwritten lines, etc.

The difficulty is further increased in the case of ancient documents, due to common problems appearing in them: presence of smear, significant background variations and uneven illumination, spots due to the humidity, and marks resulting from the ink that goes through the paper (generally called "bleed-through").

We consider TLAD as a labelling process that assigns the same label to spatially aligned units such as pixels, connected components or characteristic points [4], and also actually finds and yields the physical locations of text lines in the image; more precisely, we look for the baseline position coordinates of handwritten lines, which is defined as the fictitious straight line that follows and joins the lower part of the character bodies in a text line.

In this work we present an approach for text line analysis and detection in handwritten documents based on Hidden Markov Models (HMMs), a technique widely used in other handwritten and speech recognition tasks. It is shown that text line analysis and detection can be solved using a more formal methodology in contra-position to most of the proposed heuristic approaches found in the literature. Our approach not only provides the best position coordinates for each of the vertical page regions but also labels them, in this manner surpassing the traditional heuristic methods.

In our experiments we assess the performance of the approach (both in line analysis and detection) and review the impact of increasingly constrained "vertical layout language models" and "line type models" on text line detection and classification accuracy. Through this experimentation we also show the improvement in quality of the baselines yielded by our approach in comparison with a state-of-the-art heuristic method based on vertical projection profiles.

This document is organized as follows: In the first chapter we cover the task, motivation and our approach. On Chapter 2 we cover the theoretical concepts of our chosen model HMMs, the model language and the tool-kit used for training and decoding. We review in depth our proposed system covering the used image preprocessing techniques, feature extraction method, modelling options and evaluation measures considered in our research in Chapter 3. In Chapter 4 we describe the corpora used for training and testing, our experiment set-up and the results that helps us evaluate the different morphological and language model scenarios proposed in Chapter 3. Finally in Chapter 5 we present the conclusions extracted from our investigation, publications and future work to be covered.

1.2 Motivation

Handwritten text transcription is becoming an increasingly important task, in order to provide historians and other researchers new ways of indexing, consulting and querying the huge amounts of historic handwritten documents which are being published in on-line digital libraries.

Such access requires a digitalization of the documents of interest. This digitalization can range from a basic scanning of the pages and indexing of the documents to a full transcription of the text.

Scanning the documents and simply providing access to the images is an inadequate manner to provide access. The degradation of the pages and specialized vocabulary makes this method very difficult, therefore not useful, to researchers that have interest in the contents of the documents but are not professional transcribers.

With these premises it is clear that the only way to provide real universal access to these information sources is to develop systems that, with the aid of pattern recognition techniques, facilitates the transcription process and allows to share the results with the rest of interested parties.

Any advances on TLAD will impact significantly in the accuracy of the transcription and word spotting systems required to provide global access to the information present in legacy handwritten documents.

1.3 Related Works

Among the most popular state-of-the art methods involved in handwritten text line detection we find four main families: based on (vertical) projection profiles [6], on the Hough transform [3], the repulsive-attractive network approach [11] and finally the so-called stochastic methods [10], which combine probabilistic models such as HMMs along with dynamic programming techniques (e.g. Viterbi algorithm) to derive optimal paths between overlapping text lines.

Use of formal frameworks for segmentation tasks, as done in this work, is gaining strength as can be seen in [2] where *relative location features* are used in Conditional Random Field to perform segmentation tasks.

segmentation tasks.

It is worth noting that, most of the mentioned approaches somewhat involve heuristic adjustments of their parameters, which have to be manually tuned according to the characteristics of each task in order to obtain adequate results.

1.4 Overview of the Proposed Approach

Our approach for TLAD in handwritten documents is based on Hidden Markov Models (HMM) trained with supervised data. Training data consists of a set of detected and labelled text lines from the (kind of) documents considered.

The seminal idea of our stochastic approach can be found in [5], where manually built HMMs and a fixed ergodic grammar are used in order to detect text lines in printed text. In this work we make (formal) use of HMMs to perform TLAD in handwritten text and perform a study on the impact of the language model in the detection accuracy.

As per the classification of line segmentation techniques presented in [4] our approach for TLAD in handwritten documents can be considered to be mainly part of the stochastic family as it is based in HMMs. It is also important to note that we use projection profiles and smearing as part of our feature extraction process, hence at a lower level the approach can be considered part of these family of methods. The method due to assumptions and restrictions should be compared to those marked as suitable for straight lines with touching and overlapping components.

We extract from scanned handwritten documents a set of vertical projection profiles from different horizontal sections of the page with the intention of not only reducing the issues produced by different line lengths that are found in most projection methods but actually classifying the regions as per a set of defined page regions.

In order to minimize the issues of overlapping components we apply a smearing technique, Run Length Smoothing Algorithm (RLSA), before extracting the projection profile. We do this in order to enhance the main body of the text line; zone delimited by the base and upper lines of the text line.

Finally the projection profiles, extracted as features, are used together with the line classification labels to train Hidden Markov Models for each page "vertical region" type which are combined as per a "vertical layout language models".

Different types of "vertical layout language models" and "vertical region type models" are experimented with to review the performance impact on our approach.

1.5 Context of Application and Assumptions

For the approach presented in this paper, we assume that page images or selected image regions contain only paragraphs of single-column (roughly) parallel text lines with no images or figures.

Lines present in the page are not required to be fully separated. Although the approach allows touching and overlapping components to be present, the existence of inter-line region that presents a variation of in projection profile is assumed.

These assumptions are adequate for the large majority of handwritten documents of interest for transcription: large volumes with fairly good page structure.

1.6 Expected Outcomes/Results

The expected outcomes are:

- Primary expectation is to obtain a new method for line detection and classification based on the combination of projection, smearing and stochastic techniques.
- The Hidden Markov Model based system will be up to par or above the results of the current state of the art systems on this task without the use of heuristics.
- Obtain a simple system with the same speed for decoding as the projection techniques but with greater robustness against connected components.
- Review of the impact of the “vertical region type models” in line detection and classification.
- Analysis of the impact of the “ vertical layout language models” in line detection and classification.
- Obtain a solid base for future studies on automatic layout detection.

Bibliography

[1] Bazzi, I., Schwartz, R., and Makhoul, J. (1999). An omnifont open-vocabulary OCR system for English and Arabic. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(6):495–504.

[2] Cruz, F. and Terrades., O. R. (2012). Document segmentation using relative location features. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012).*

[3] Likforman-Sulem, L., Hanimyan, A., and Faure, C. (1995). A hough based algorithm for extracting text lines in handwritten documents. *Document Analysis and Recognition, International Conference on*, 2:774.

[4] Likforman-Sulem, L., Zahour, A., and Taconet, B. (2007). Text line segmentation of historical documents: a survey. *Int. J. Doc. Anal. Recognit.*, 9:123–138.

[5] Lu, Z., Schwartz, R., and Raphael, C. (2000). Script-independent, hmm-based text line finding for ocr. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 4, pages 551 –554 vol.4.

[6] Sánchez-Cortina, I., Serrano, N., Sanchis, A., and Juan, A. (2012). A prototype for interactive speech transcription balancing error and supervision effort. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, pages 325–326.

[7] Toselli, A. H., Juan, A., Keysers, D., González, J., Salvador, I., H. Ney, Vidal, E., and Casacuberta, F. (2004). Integrated Handwriting Recognition and Interpretation using Finite-State Models. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 18(4):519–539.

[8] Toselli, A. H., Romero, V., Pastor, M., and Vidal, E. (2009). Multimodal interactive transcription of text images. *Pattern Recognition*, 43(5):1824–1825.

[9] Toselli, A. H., Romero, V., and Vidal, E. (2011). *Language Technology for Cultural Heritage*, chapter Alignment between Text Images and their Transcripts for Handwritten Documents., pages 23–37. Theory and Applications of Natural Language Processing. Springer,. Caroline Sporleder, Antal van den Bosch y Kalliopi Zervanou (Eds.).

[10] Vinciarelli, A., Bengio, S., and Bunke, H. (2004). Off-line recognition of unconstrained handwritten texts using hmms and statistical language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):709–720.

[11] Öztop, E., Mülayim, A., Atalay, V., and Yarman-Vural, F. (1999). Repulsive attractive network for baseline extraction on document images. *Signal Processing*, 75(1):1 – 10.

CHAPTER 2

FUNDAMENTS

Chapter Outline

2.1 Hidden Marcov Models (HMMs)

A Hidden Markov Models (HMM) is an statistical Markov model where only the emissions and not the states are visible. A HMM can be represented by:

A finite set of states each of which is associated with a continuous probability distribution that defines or rules the "emissions" in that state.

Transitions probability which governs the transitions among the states of the HMM.

HMMs are considered the most successful model used for Automatic Speech Recognition (ASR) due to the results obtained with them during the past decades. This successes are due to the models ability to represent the problem and its sequential nature in a formal way , hence mathematically tractable, the discrete time sequences of the extracted acoustic feature vectors.

HTR shares one major similarity with ASR as the discrete time sequences that define the continuous writing can be considered also as emissions from an HMM. For HTR the observed emissions represent line-image features and point coordinates of the handwritten pen strokes. Due to this similarity and the success of HMM for the ASR task the application of these statistical models has gained popularity for the resolution of HTR tasks.

For layout detection the same similarity can be considered as the vertical regions that define the page can be considered as emissions from an HMM.

2.1.1 Definition

One possible classification of HMMs can be performed according to the nature of the observed emissions [2][1]

- If the observed emissions are represented by a vector of symbols in a finite alphabet the HMM can be considered **discrete**
- When emissions are vectors of reals the HMM is defined as **continuous**
- **semi-continuous** is used when the emissions are of a discrete nature but are modelled using continuous probability density functions.

Since in our research we will work only with continuous HMMs we will provide a summarized formal definition of this kind of HMM using the notation represented in [4]. Following assumptions are considered for our continuous HMMs:

- Emissions are only performed in states and not in transitions
- An additional initial state similar to the end state, which can not perform emissions, has been defined

Both in ASR and HTR, HMMs are used to compute the probability of the input signal represented as a sequence of feature vectors. Let be $\mathbf{x} = \boldsymbol{x}_1\boldsymbol{x}_2 ... \boldsymbol{x}_T$, a sequence of real vectors, a HMM M approximates the probability of this sequence; that is[a]:

$$\Pr(\mathbf{x}) \approx P_M(\mathbf{x}) \tag{2.1}$$

Formally, a continuous HMM is a finite state machine defined by the sextuple (Q, I, F, X, a, b) where:

- Q is a finite set of states. In order to avoid confusions with the indexation of the different states, we denote the states of the model as $q_0, ..., q_{|Q|-1}$, whereas a sequence of states that generates the vector sequence $\mathbf{x} = \boldsymbol{x}_1\boldsymbol{x}_2 ... \boldsymbol{x}_T$ will be denoted as $z_1 z_2 ... z_T$.
- I is the initial state, an element of Q: $I \in Q$. $I = q_0$
- F is the final state, an element of Q: $F \in Q$. $F = q_{|Q|-1}$
- X is a real d-dimensional space of observations: $X \subseteq \Re^d$.
- a is the state-transition probability function[b]:

  $$a(q_i, q_j) = P(z_{t+1} = q_j | z_t = q_i) \quad q_i \in (Q - \{F\}), \quad q_j \in (Q - \{I\})$$

  Transition probabilities should satisfy $a(q_i, q_j) \geq 0$ and

  $$\sum_{q_j \in (Q - \{I\})} a(q_i, q_j) = 1 \quad \forall q_i \in (Q - \{F\})$$

- b is a probability distribution function[c]:

  $$b(q_i, \boldsymbol{x}) = P(\boldsymbol{x}_t = \boldsymbol{x} | z_t = q_i) \quad q_i \in (Q - \{I, F\}), \quad \boldsymbol{x} \in X$$

  The following stochastic constraints must be satisfied: $b(q_i, \boldsymbol{x}) \geq 0$ and

  $$\int_{\boldsymbol{x} \in X} b(q_i, \boldsymbol{x}) d\boldsymbol{x} = 1 \quad \forall q_i \in (Q - \{I, F\})$$

As the observations are continuous we must therefore use a continuous probability density function. In this case it is defined as a weighted sum of G Gaussian distributions:

$$b(q_j, \boldsymbol{x}) = \sum_{g=1}^{G} c_{jg} b_g(q_j, \boldsymbol{x})$$

where,

$$b_g(q_j, \boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_{jg}|}} e^{(-\frac{1}{2}(\boldsymbol{x} - \mu'_{jg}) \Sigma_{jg}^{-1} (\boldsymbol{x} - \mu_{jg}))}$$

[a]"True" probabilities are written as $\Pr(\ldots)$, in contrast with model approximations such as $P_M(z \mid \ldots)$ which, to simplify notation, will be denoted as $P(z \mid \ldots)$ whenever M can be understood

[b] $z_t = q_i$ means that the HMM is in the state q_i at time t

[c] $\boldsymbol{x}_t = \boldsymbol{x}$ means that the HMM in the state z_t generates $\boldsymbol{x}$ at time t

- μ_{jg} is the mean vector for the component g of the state q_j
- Σ_{jg} is the covariance matrix for the component g of the state q_j
- c_{jg} is the weighting coefficient for the component g of the state q_j, and should satisfy the stochastic constrain $c_{jg} \geq 0$ and
$$\sum_{g=1}^{G} c_{jg} = 1$$

For the sake of mathematical and computational tractability, the following assumptions are made in the theory of HMMs:

1. **The Markov assumption.** As given in the definition of HMMs, transition probabilities are defined as: $a(q_i, q_j) = P(z_{t+1} = q_j | z_t = q_i)$. In other words it is assumed that the next state is dependent only upon the current state; that is,
$$\Pr(z_{t+1}|z_1...z_t) \approx P(z_{t+1}|z_t)$$
Which is called the Markov assumption and when applying it to the generalized HMM the resulting model becomes actually a first order HMM.
2. **The stationary assumption.** Assumes that state transition probabilities are independent of the actual time at which the transitions take place. Mathematically,
$$P(z_{t_1+1} = q_j | z_{t_1} = q_i) = P(z_{t_2+1} = q_j | z_{t_2} = q_i)$$
for any t_1 and t_2
3. **The output independence assumption.** The probability distribution function is defined as: $b(q_i, \boldsymbol{x}) = p(\boldsymbol{x}_t = \boldsymbol{x} | z_t = q_i)$. This means that the current output (observation) is statistically independent of the previous outputs (observations) and it only depends of the current state; that is,
$$\Pr(\boldsymbol{x}_t|\boldsymbol{x}_1...\boldsymbol{x}_{t-1}, z_1...z_t) \approx P(\boldsymbol{x}_t|z_t)$$

2.1.2 Basic algorithms for HMMs

Once we have a HMM, there are three problems of interest. The evaluation problem, the decoding problem and the learning problem.

- The Evaluation Problem. This problem consist in computing the probability $P(\mathbf{x})$; that is, the probability that the observations are generated by the model.
- The Decoding Problem. Given a HMM and a sequence of observations **x**, the problem is to find the most likely state sequence in the model which produced the observations. In other words, the problem consists on finding the hidden part of the HMM.

- The Learning Problem. Given a HMM and a sequence of observations **x**, how should we adjust the model parameters in order to maximize the probability $P(\mathbf{x})$.

To simplify the notation, in the next sections, $a(q_i, q_j)$ will be written as a_{ij} and $b(q_i, x)$ as $b_i(x)$[d].

The Evaluation Problem and the Forward and Backward Algorithms

Let **x** be a sequence of real vectors and $Z = \{\mathbf{z} = z_1 z_2 ... z_T : z_k = q_i \in (Q - \{I, F\}), 1 \leq i \leq |Q| - 2\}$ a set of state sequences associated with the vector sequence **x**. Then, the probability that **x** be generated by the HMM is:

$$P(\mathbf{x}) = \sum_{\mathbf{z} \in Z} \left(\prod_{i=1}^{T} a_{z_{i-1} z_i} b_{z_i}(\boldsymbol{x}_i) \right) a_{z_T F}$$

where z_0 is the initial state I: $z_0 = q_0 = I$.

This calculation involves a number of operations that is in the order of N^T, where N is the number of states of the model excluding the initial state, $N = |Q| - 1$ ($Q = \{q_0 = I, q_1, ..., q_{N-1}, q_N = F\}$), and T is the number of vectors of the sequence. This is very large even if the length of the sequence, T is moderate. Hence, for practical reasons, we must look for another method to perform this calculation.

The **Forward** algorithm is an efficient algorithm which computes $P(\mathbf{x})$. The time complexity order of this algorithm is: $O(|Q|^2 \cdot T)$; however, if we further restrict the HMM topology to using a left-to-right HMM the complexity falls to $O(|Q| \cdot T)$. In the left-to-right HMM topology a transition between two states $q_i, q_j \in Q$ from the HMM, it is only possible if $j \geq i$.

The forward function $\alpha_j(t)$ for $0 < j < N$, is defined as the probability of the partial observation sequence $x_1 x_2 ... x_t$, when it terminates at the state j. Mathematically, $\alpha_j(t) = P(\mathbf{x}_1^t, q_j)$ and it can be expressed in the following recursive manner:

$$\alpha_j(t) = \begin{cases} a_{0j} b_j(x_1) & t = 1 \\ \left(\sum_{i=1}^{N-1} \alpha_i(t-1) a_{ij} \right) b_j(\boldsymbol{x}_t) & 1 < t \leq T \end{cases}$$

with the initial condition that $\alpha_0(1) = 1$. Using this recursion we can calculate the probability that the sequence **x** be emitted by the model M as:

$$P(\mathbf{x}) = P(\mathbf{x}_1^T) = \alpha_N(T) = \sum_{i=1}^{N-1} \alpha_i(T) a_{iN}$$

[d]From now on, any kind of subsequence will be represented as $l_i ... l_j$ or as $\mathbf{l}_i^j$, whenever it is convenient.

In a similar way we can define the **Backward** function $\beta_i(t)$ for $0 < i < N$, as the probability of the partial observation sequence $x_{t+1}x_{t+2}...x_T$, given that the current state is i. Mathematically, $\beta_i(t) = P(\mathbf{x}_{t+1}^T|q_i)$ and it can be expressed on a recursive way:

$$\beta_i(t) = \begin{cases} a_{iN} & t = T \\ \sum_{j=1}^{N-1} a_{ij} b_j(\boldsymbol{x}_{t+1}) \beta_j(t+1) & 1 \leq t < T \end{cases}$$

with the initial condition that $\beta_N(T) = 1$. Using this recursion the probability that the sequence $\mathbf{x}$ be emitted by the model M can be calculated as:

$$P(\mathbf{x}) = P(\mathbf{x}_1^T) = \beta_0(1) = \sum_{j=1}^{N-1} a_{0j} b_j(x_1) \beta_j(1)$$

As in the forward algorithm the time complexity is: $O(|Q|^2 \cdot T)$, and using a left-to-right HMM the complexity falls to $O(|Q| \cdot T)$.

The Decoding Problem and the Viterbi Algorithm

In this case we want to find the most likely state sequence, $\mathbf{z} = z_1 z_2 ... z_T$, for a given sequence of observations, $\mathbf{x}$. The algorithm used here is commonly known as the Viterbi algorithm, which maximizes the joint probability of the observations and all possible sequence of states; that is $\max_{\mathbf{z}} P(\mathbf{x}, \mathbf{z})$. This algorithm is similar to the forward algorithm, but replacing the sum by the dominating term.

$$v_j(t) = \begin{cases} a_{0j} b_j(x_1) & t = 1 \\ \left(\max_{i \in [1, N-1]} v_i(t-1) a_{ij}\right) b_j(\boldsymbol{x}_t) & 1 < t \leq T \end{cases}$$

with the condition that $v_0(1) = 1$. $v_N(T)$ is the probability $\max_{\mathbf{z}} P(\mathbf{x}, \mathbf{z})$ and using this recursion it can be calculated as:

$$v_N(T) = \max_{i \in [1, N-1]} v_i(T) a_{iN} \leq \sum_{i=1}^{N-1} \alpha_i(T) a_{iN} = \alpha_N(T)$$

The time complexity of the Viterbi algorithm is: $O(|Q|^2 \cdot T)$, and using a left-to-right HMM the complexity falls to $O(|Q| \cdot T)$.

The Learning Problem and the Baum-Welch Algorithm

The learning problem is how to adjust the HMM parameters ($a_{ij}, b_i(x), c_{jg}, \mu_{jg}$ and Σ_{jg}), so that a given set of observations (called training set) is generated by the model with maximum likelihood.

The Baum-Welch algorithm [3] (also known as Forward-Backward algorithm), is used to find these unknown parameters. It is an expectation-maximization (EM) algorithm.

Let $E = \{\mathbf{x}_r = \boldsymbol{x}_{r1}\boldsymbol{x}_{r2}...\boldsymbol{x}_{rT_r} : \boldsymbol{x}_{rk} \in X, 1 \leq k \leq T_r \wedge 1 \leq r \leq R\}$ a set of R vector sequences, used to adjust the HMM parameters. The basic formula to estimate the state-transition probability a_{ij} is:

$$\hat{a}_{ij} = \frac{\sum_{r=1}^{R} \frac{1}{P_r} \sum_{t=1}^{T_r-1} \alpha_i^r(t) a_{ij} b_j(\boldsymbol{x}_{rt+1}) \beta_j^r(t+1)}{\sum_{r=1}^{R} \frac{1}{P_r} \sum_{t=1}^{T_r} \alpha_i^r(t) \beta_i^r(t)}$$

where $0 < i < N$, $0 < j < N$ and $P_r = P(\mathbf{x}_r)$ is the total probability of the sample r from the set E.

If the probability density function of each state on the HMM is approximated by a weighted sum of G Gaussian distributions we must find the unknown parameters c_{jg}, μ_{jg} and Σ_{jg}. With this purpose we define $L_{jg}^r(t)$ as the probability that the vector $\boldsymbol{x}_{rt} \in \Re^d$ be generated by the Gaussian component g in the q_j state:

$$L_{jg}^r(t) = \frac{1}{P_r} U_j^r(t) c_{jg} b_{jg}(\boldsymbol{x}_{rt}) \beta_j^r(t)$$

where

$$U_j^r(t) = \begin{cases} a_{0j} & if \quad t = 1 \\ \sum_{i=1}^{N-1} \alpha_i^r(t-1) a_{ij} & otherwise \end{cases}$$

Taking into account the previous definitions, the parameters c_{jg}, μ_{jg} and Σ_{jg} can be estimated as:

$$\hat{\mu}_{jg} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_{jg}^r(t) \boldsymbol{x}_{rt}}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_{jg}^r(t)}$$

$$\hat{\Sigma}_{jg} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_{jg}^r(t) (\boldsymbol{x}_{rt} - \hat{\mu}_{jg})(\boldsymbol{x}_{rt} - \hat{\mu}_{jg})'}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_{jg}^r(t)}$$

$$c_{jg} = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_{jg}^r(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T_r} L_j^r(t)}$$

The time complexity of one iteration of the Baum-Welch algorithm is: $O(R \cdot |Q|^2 \cdot T)$; however, using a left-to-right HMM the complexity falls to $O(R \cdot |Q| \cdot T)$. This algorithm is iterated until some convergence criterion is reached.

Sometimes, it is necessary to have a composition of C HMM joined sequentially, for example in the case of the different vertical regions that conform a page. In this case, the "embedded training Baum-Welch" algorithm, which re-estimates the parameters of the composition of C sequentially concatenated HMMs, can be used. This algorithm enables to train the HMM without any prior segmentation of the training page images into vertical regions. In [4] we can find all the formula to compute the unknown parameters in this case but more specifically for the HTR case (lines and morpheme).

2.2 Language models: N-grams

Language models (LMs) are usually used to model text properties, like syntax and semantic, independently from the character morphology modelled by HMMs. They are used in many natural language processing applications such as speech recognition, machine translation or handwritten recognition. These models can be used to predict the next word in a word sequence.

In our research LMs are used in order to model the structure of a page as described by the composition of the different regions / text sections that can compose it. Language models assign a probability to a sequence of regions $\mathbf{w} = w_1, w_2, ..., w_l$, which can be expressed as:

$$\Pr(\mathbf{w}) = \Pr(w_1) \cdot \prod_{i=2}^{l} \Pr(w_i|\mathbf{w}_1^{i-1})$$

where $\Pr(w_i|\mathbf{w}_1^{i-1})$ is the probability of the region w_i when we have already seen the sequence of regions $w_1...w_{i-1}$. The sequence of regions prior to w_i is called history.

In practice ,for HTR, estimating the probability of sequences can become difficult since sentences can be arbitrarily long and hence many sequences are not observed during LM training. It is necessary to note that for a vocabulary with $|V|$ different words, the number of different histories is $|V|^{i-1}$. So, the estimation of $\Pr(\mathbf{w})$ can be unworkable. For that reason these models are often approximated using smoothed n-gram models, which obtains surprisingly good performance although they only captures short term dependencies.

In the case of layout detection, considering the allowed region types, it is more unusual as pages usually have a similar number of lines and samples of transitions from one region to the rest can be found.

An n-gram defines a function: $\Phi_n : V^* \to V^{n-1}$ in which, all sequences finishing with the same $n-1$ words belong to the same equivalence class. Now, $\Pr(\mathbf{w})$ can be approximated as:

$$\Pr(\mathbf{w}) \approx \prod_{i=1}^{l} P(w_i|\Phi_n(\mathbf{w}_1^{i-1})) = \prod_{i=1}^{l} P(w_i|\mathbf{w}_{i-n+1}^{i-1}) \tag{2.2}$$

Owing to the fact that $i - n \leq 0$ for the first $n-1$ words in $\mathbf{w}$, Eq. (2.2) must be written as:

$$\Pr(\mathbf{w}) \approx P(w_1) \cdot \prod_{i=2}^{n-1} P(w_i|\mathbf{w}_1^{i-1}) \cdot \prod_{i=n}^{l} P(w_i|\mathbf{w}_{i-n+1}^{i-1})) \tag{2.3}$$

Given a vocabulary V and a transcribed training data or text corpora represented by $\mathbf{w} = w_1w_2...w_l$, the estimated probability of the word $v \in V$, having seen a sequence of $n-1$ words $\mathbf{v} \in V^{n-1}$, is computed as:

$$P(v|\mathbf{v}) = \frac{C(\mathbf{v}v)}{C(\mathbf{v})}$$

where $C(\mathbf{v})$ is the number of times that the sequence $\mathbf{v}$ has appeared in the training sequence $\mathbf{w}$. This is a maximum likelihood (ML) estimate.

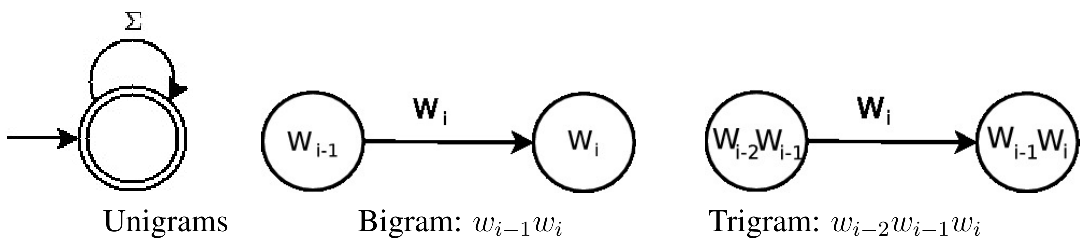

Figure 2.1: Examples of n-grams represented using a SFSA.

n-grams modelled by a stochastic finite state automaton

Along this work, stochastic finite state automata (SFSA) are often used to represent HMMs, lexical models and language models. Thanks to the homogeneous finite-state nature of all these models, they can be easily integrated into a single global finite state model. A n-gram can be represented using a SFSA [5, 6], defined as a sextuple $A = (Q, V, \delta, q_0, P, F)$, where:

- V is non-empty finite set of symbols
- $Q \subseteq V^{n-1} \cup q_0$ is a finite, not-empty set of states. Each state is defined using the vocabulary symbols V as $q = (v_{i-n+1}...v_{i-2}v_{i-1}) \in Q$
- $\delta \subset Q \times V \times Q$ is the state-transition function. A transition is denoted as:

  $$(v_{i-n+1}...v_{i-2}v_{i-1}, v, v_{i-n+2}...v_{i-1}v)$$

  where $(v_{i-n+1}...v_{i-2}v_{i-1}) \in Q$, $(v_{i-n+2}...v_{i-1}v) \in Q$, and $v \in V$
- q_0 is the initial state $(q_0 \in Q)$
- $P : \delta \rightarrow \Re^+$ is the probability transition function. We are using deterministic SFSA, so each transition is identified with only the source state $q \in V^{n-1}$ and the transition symbol $v \in V$. Therefore, $P(q, v, q') = P(v|q)$
- $F : Q \rightarrow \Re^+$ is the final state probability function.

2.3 HTK ToolKit

The Hidden Markov Model Tool-kit (HTK) [7] is an Open Source tool-kit developed and maintained at the Cambridge University Engineering Department (CUED). Development started in 1989 by the Speech Vision and Robotics Group as a set of modules developed in C to perform speech recognition research using HMMs as the statistical model.

During its lifetime the HTK licensing and distribution form and owning company has had changes. As of 1999 Microsoft through the acquisition Entropic Research Laboratories (ERL) had the license rights to HTK. As of 2000 Microsoft has licensed back HTK to CUED so that it can maintain it and distribute it.

In our research we have used HTK (v3.4) in order to:

- Build and train HMMs for the specific page regions.
- Test the resulting HMMs to review its performance.

The software and manual for usage can be found in http://htk.eng.cam.ac.uk/

Bibliography

[1] Jelinek, F. (1998). *Statistical Methods for Speech Recognition*. MIT Press.

[2] Lee, K. (1989). *Automatic speech recognition: the development of the SPHINX system*. Kluwer international series in engineering and computer science. Kluwer Academic Publishers.

[3] Rabiner, L. R. and Juang, B. H. (1993). *Fundamentals of Speech Recognition*. Prentice-Hall, Englewood Cliffs, New Jersey, USA.

[4] Toselli, A. H. (2004). *Reconocimiento de Texto Manuscrito Continuo*. PhD thesis, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, Valencia (Spain). Advisor(s): Dr. E. Vidal and Dr. A. Juan (in Spanish).

[5] Vidal, E., Thollard, F., C. de la Higuera, F. C., and Carrasco, R. (2005a). Probabilistic finite-state machines - part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025.

[6] Vidal, E., Thollard, F., C. de la Higuera, F. C., and Carrasco, R. (2005b). Probabilistic finite-state machines - part II. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1025–1039.

[7] Young, S. J., Evermann, G., Gales, M. J. F., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., and Woodland, P. C. (2006). *The HTK Book, version 3.4*. Cambridge University Engineering Department, Cambridge, UK.

CHAPTER 3

HMM-BASED TEXT LINE DETECTION SYSTEM

Chapter Outline

3.1 System Architecture

The flow diagram of Fig. 3.1 displays the overall process of the proposed handwritten text line analysis and detection approach.

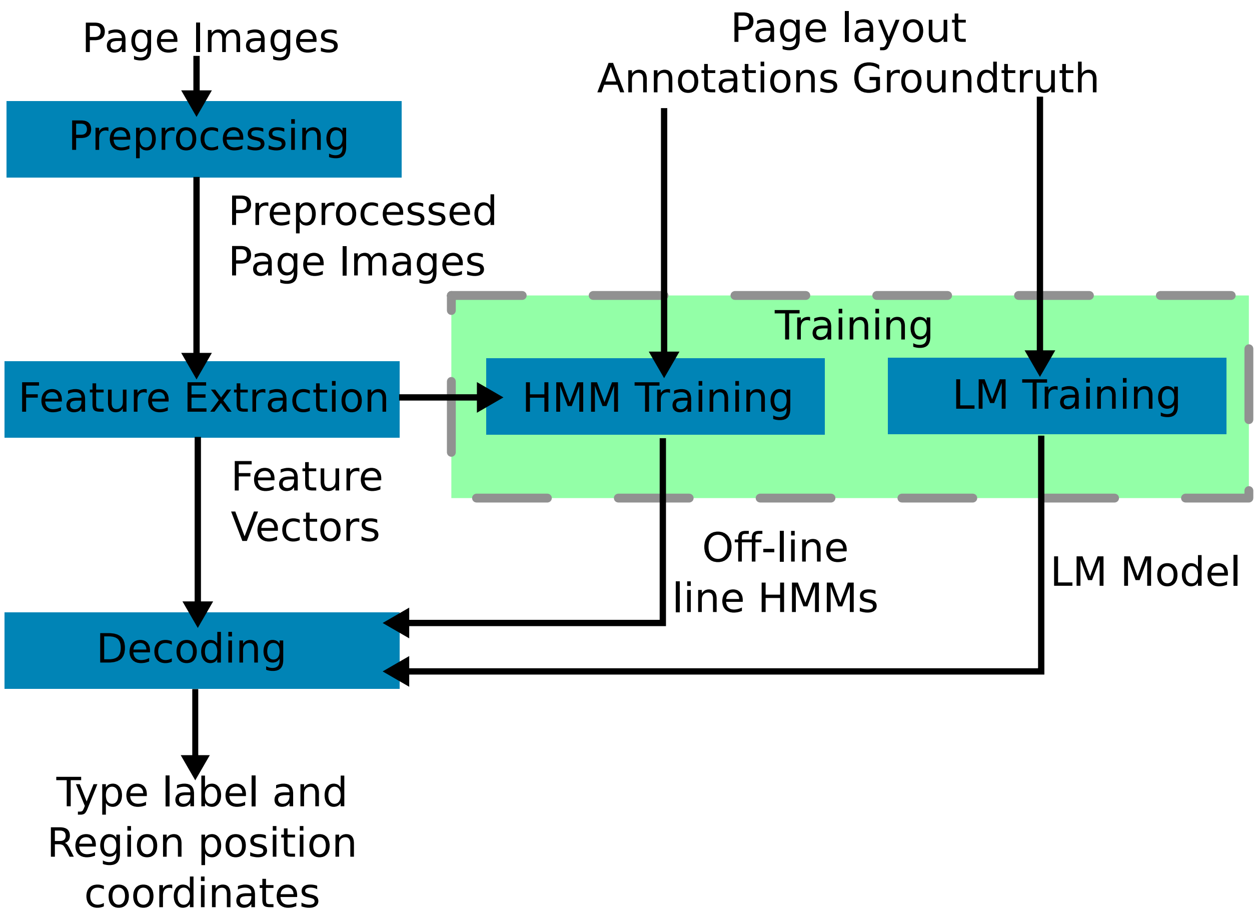

Figure 3.1: Global scheme of the handwritten text line detection process.

The system is composed of four different phases:

Image Preprocessing: In this phase the pages are processed individually in order to discard/reduce any error or noise present in the original page image.

Feature extraction: Phase where a predefined set of features are extracted from the preprocessed image, that must represent adequately the scanned image for the purpose of our task.

Training: In this phase specific Hidden Markov Models and Language Models are trained for an specific document.

Decoding: In this final stage, the system yields a set of baseline coordinates and vertical region type labels given an input feature vector set of an specific page.

In the following sections we will review the modelling required for the training phase, the methods applied in the image processing stage and the manner in which the feature extraction is performed.

3.2 Modelling

3.2.1 Statistical Framework Contextualization

Similarly to how the statistic framework of automatic speech and handwritten text recognition (ASR, HTR) is established, the handwritten text line detection problem can be also formulated as the problem of finding a most likely line label sequence hypothesis, $\hat{\mathbf{h}} = \langle \hat{h}_1, \hat{h}_2, \ldots, \hat{h}_n \rangle$, for a given handwritten page image (or selected region image) represented by an observation sequence[a] $\mathbf{o} = \langle \vec{o_1}, \vec{o_2}, \ldots, \vec{o_L} \rangle$, that is:

$$\hat{\mathbf{h}} = \arg\max_{\mathbf{h}} P(\mathbf{h} \mid \mathbf{o}) \tag{3.1}$$

Using the Bayes' rule we can decompose the probability $P(\mathbf{h} \mid \mathbf{o})$ into two terms:

$$\hat{\mathbf{h}} = \arg\max_{\mathbf{h}} P(\mathbf{o} \mid \mathbf{h}) \cdot P(\mathbf{h}) \tag{3.2}$$

In the jargon of ASR or HTR these terms represent the morphological and syntactic knowledge level respectively, where $P(\mathbf{o} \mid \mathbf{h})$ is typically approximated by HMMs, while $P(\mathbf{h})$ by an N-gram language model (LM).

In this work, we are interested not only in detecting and labelling the text lines in a given (page) image, but also their exact physical locations on it. In this sense, by solving Eq. (3.2), such physical locations are determined by the optimal subsequences of $\mathbf{o}$ aligned with each of the detected text lines $h_1, h_2, \ldots, h_n$. These optimal subsequences are implicit or "hidden" in Eq. (3.2), which can be rewritten as:

$$\hat{\mathbf{h}} = \arg\max_{\mathbf{h}} \sum_{\mathbf{b}} P(\mathbf{o}, \mathbf{b} \mid \mathbf{h}) \cdot P(\mathbf{h})$$

where $\mathbf{b}$ is an *alignment*; that is, an ordered sequence of $n+1$ marks $\langle\, b_0, b_1, \ldots, b_n \rangle$, used to demarcate the subsequences belonging to each text line. The marks b_0 and b_n always point out to the first and last components of $\mathbf{o}$ (see Fig. 3.2). Now, approximating the sum in (3.3) by the dominant term, $\max_{\mathbf{b}} P(\mathbf{o}, \mathbf{b} \mid \mathbf{h})$:

$$(\hat{\mathbf{b}}, \hat{\mathbf{h}}) \approx \arg\max_{\mathbf{b},\mathbf{h}} P(\mathbf{h}) \cdot P(\mathbf{o}, \mathbf{b} \mid \mathbf{h}) \tag{3.3}$$

where $\hat{\mathbf{b}}$ is the optimal alignment. Eq. (3.3) can be expanded to,

$$(\hat{\mathbf{b}}, \hat{\mathbf{h}}) = \arg\max_{\mathbf{b},\mathbf{h}} P(\mathbf{h}) \cdot P(o_{b_0}^{b_1} \mid \mathbf{h})\, P(o_{b_1}^{b_2} \mid o_{b_0}^{b_1}, \mathbf{h}) \ldots P(o_{b_{n-1}}^{b_n} \mid o_{b_0}^{b_{n-1}}, \mathbf{h}) \tag{3.4}$$

[a]Henceforward, in the context of this formal framework, each time it is mentioned image of *page or selected text*, we are implicitly referring to its input feature vector sequence "$\mathbf{o}$" describing it. See details in section 3.4.

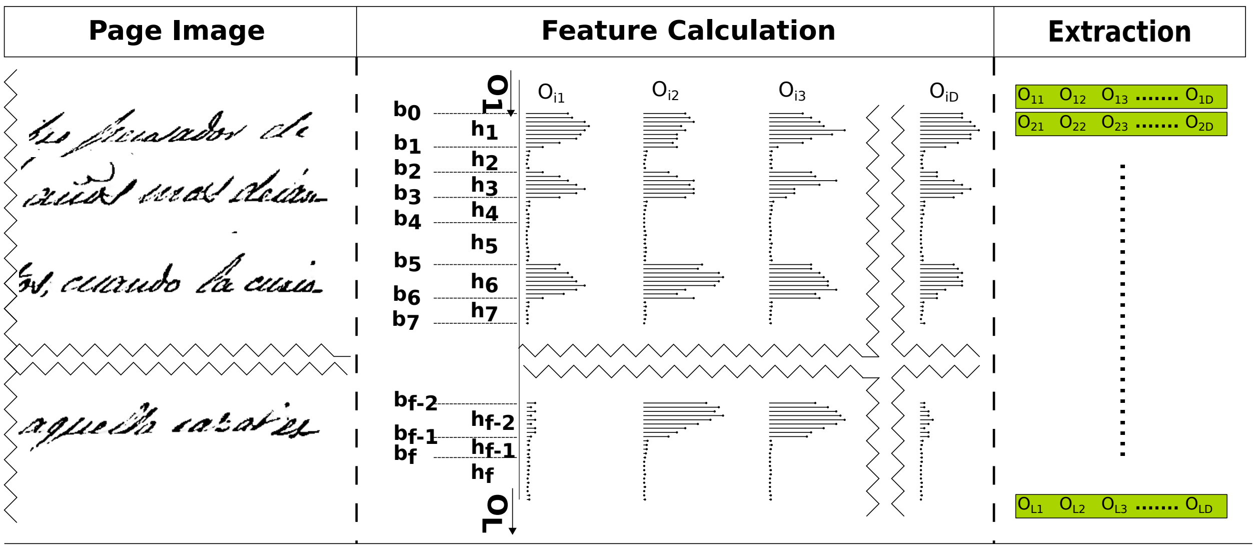

Figure 3.2: Schematic representation for a page image extraction of L feature vectors of D components. See details in section 3.4.

Assuming that each subsequence $o_{b_{i-1}}^{b_i}$ is independent from $o_{b_0}^{b_1}, \ldots, o_{b_{i-2}}^{b_{i-1}}$, and it also depends only of h_i, Eq. (3.4) can be rewritten as,

$$(\hat{\mathbf{b}}, \hat{\mathbf{h}}) \approx \arg\max_{\mathbf{b},\mathbf{h}} P(\mathbf{h}) \cdot P(o_{b_0}^{b_1}|h_1) \ldots P(o_{b_{n-1}}^{b_n}|h_n) \tag{3.5}$$

which is optimally solved by using the Viterbi search algorithm [2].

3.2.2 Morphological Models

Our line detection approach is based on two modelling levels: morphological and syntactical. The morphological level, expressed as the $P(\mathbf{o} \mid \mathbf{h})$ term (see Eq.(3.2) and Eq.(3.5)), is modelled by using HMMs and is in charge of explaining the different vertical line regions classes that appear on the input images along its vertical direction. In our line detection approach six different kinds of vertical regions are defined:

Normal text Line-region (NL): Region occupied by the main body of a normal handwritten text line.

Short text Line-region (SL): Identifies the main body of a text line that does not use the full width of the page .e.g: end of paragraphs, section headers, etc.

Paragraph text Line-region (PL): Main body of a text line that presents an indentation at the left hand side of the page. Appears mainly at the beginning of paragraphs.

Inter Line-region (IL): Defined as the region found within two consecutive text lines, characterized by being crossed by the ascenders and descenders belonging to the adjacent text lines.

Blank Line-region (BL): Large rectangular region of blank space usually found at the start and end of page image (top and bottom margins).

Non-text Line-region (NT): Stands for everything which does not belong to any of the other regions.

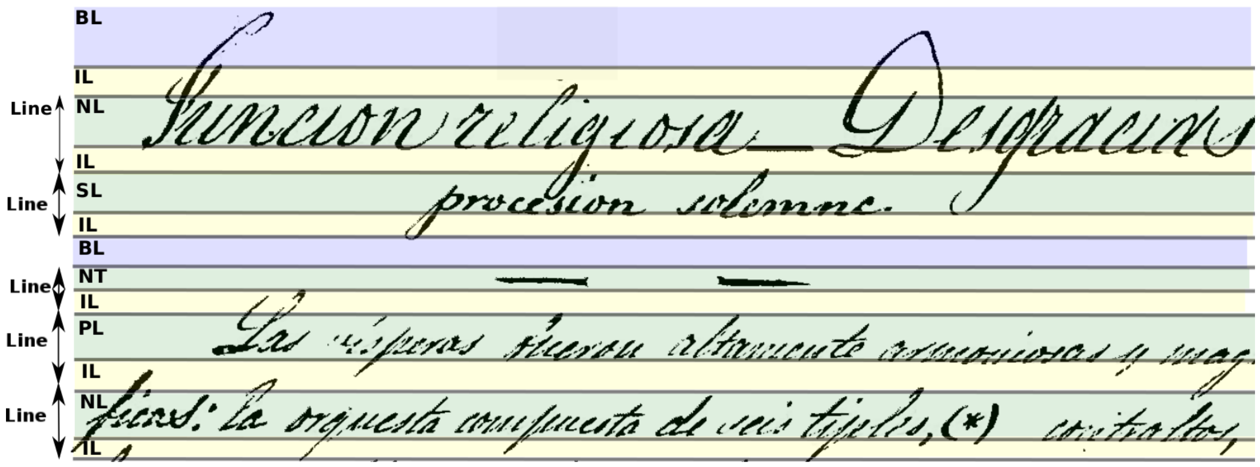

Figure 3.3: Image shows the different vertical region types: we can observe the difference in length and indentation of the NL,PL and SL types, the difference in height of of BL and IL and a sample NT mark.

We model each of these regions by an HMM which is trained with instances of such regions. Basically, each line-region HMM is a stochastic finite-state device that models the succession of feature vectors extracted from instances of the specific line-region images. In turn, each HMM state generates feature vectors following an adequate parametric probabilistic law; typically a mixture of Gaussian densities.

We review two type of HMM topologies:

Strictly Linear Models

We use this term to define a strict left to right HMM where from each state we can only transition to itself or the next one.

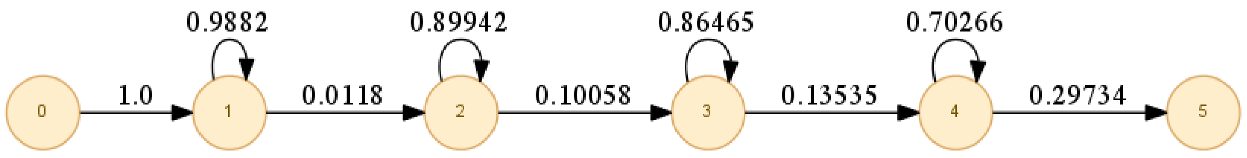

Figure 3.4: Example of a trained 6 state strictly linear model.

Ranged Linear Models

Left to right model that is constructed in order to force a hard limit on the minimum number of states it must transition and a soft limit on the maximum it accepts. This model can be divided into three stages:

- Minimum transitions stage - where the states are only allowed to transition to the next state hence setting a hard limit on the minimum number of frames.
- Maximum variable transitions stage - where the states are allowed to transition to the next state or the final state of the model.
- Final state stage - the final state allows a loop to itself in order to accommodate larger samples.

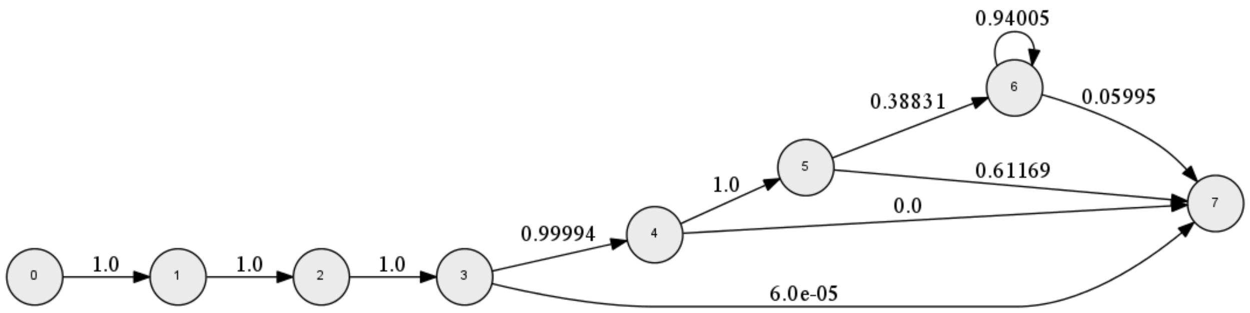

Figure 3.5: Example of a trained 8 state ranged linear model. This model forces a minimum of 3 transitions to be performed.

Once an HMM "topology" (number of states and structure) has been adopted, the model parameters can be easily trained from instances (sequences of features vectors) of full images containing a sequence of line-regions (without any kind of segmentation) accompanied by the reference labels of these images that correspond to the actual sequence of line-region classes. This training process is carried out using a well known instance of the EM algorithm called forward-backward or Baum-Welch re-estimation [2].

3.2.3 Language Model

The syntactic modelling level , expressed as the $P(\mathbf{h})$ term (see Eq.(3.2) and Eq.(3.5)), is responsible for defining the way that the different line regions can be concatenated in order to produce a valid page structure. It is worth noting that at this level, NL, PL, SL and NT line regions are always forced to be followed by IL region: NL+IL, PL+IL, SL+IL and NT+IL.

We can also use the LM to impose restrictions about the minimum or maximum number of line-regions to be detected. The LM for our text line detection approach, is implemented as a stochastic finite state grammar (SFSG) which recognizes valid sequences of elements (line regions). In our research we considered the following language models: *prior* (PRI), *conditional* (CND) and *line-number constrained* (LN-C) language models, each represented by topological different SFSGs.

The PRI LM transition probabilities are estimated from the training set as the fraction of the number of appearances of each line region label over the whole count of labels. An example can be seen in Fig. 3.6.

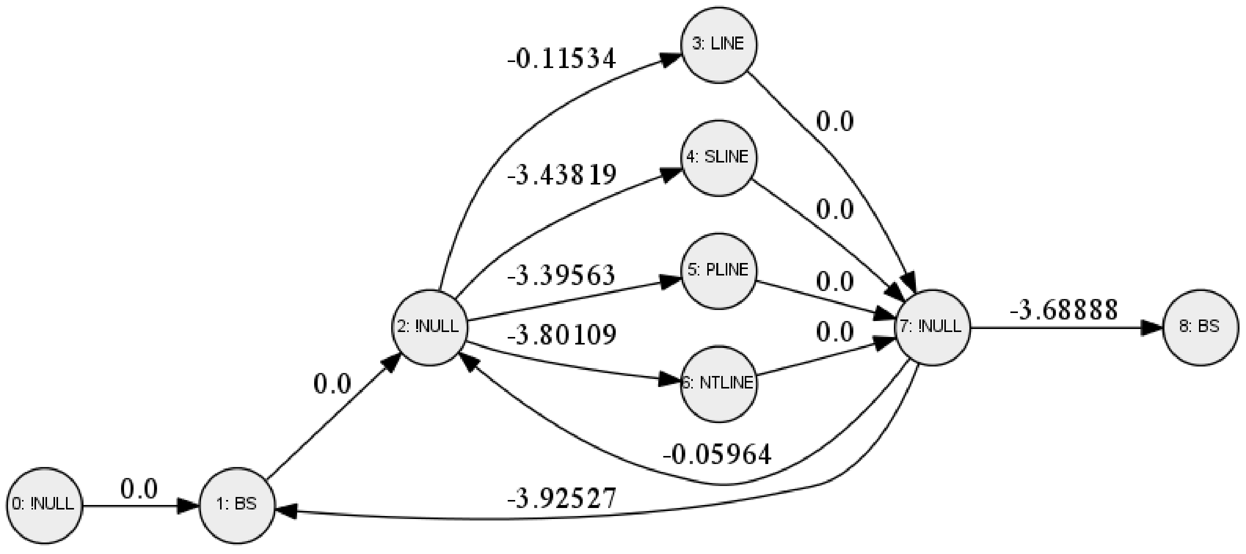

Figure 3.6: Example of a Prior Language Model..

The CND LM also considers context of the previous line region label in order to perform the estimation. An example can be seen in Fig. 3.7.

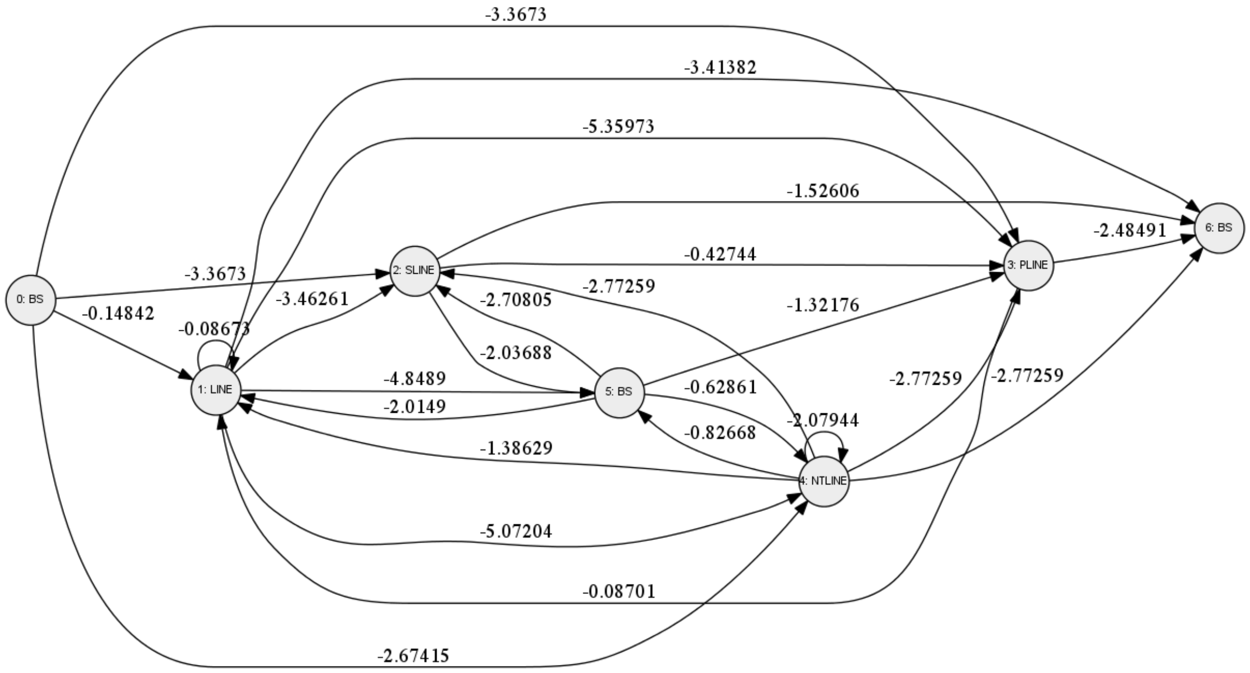

Figure 3.7: Example of a Conditional Language Model. An initial ergodic model was considered and the probabilities where recalculated dropping all non-used transitions.

It is important to note that the way the PRI and COND LMs are built somewhat resemble the uni-gram and bi-gram LMs calculations, except no smoothing strategy is implemented here.

Finally we defined for each test page a LN-C LM, which also uses the CND LM probabilities to populate the model, that enforces a total number of possible line-regions (line or blank space) to detect as per the number of reference line-region labels of that test page. LN-C is conceived for its utilization in (parts of) documents or document collections that present a homogeneous number of lines per page. An example of this LM can be seen in Fig. 3.8.

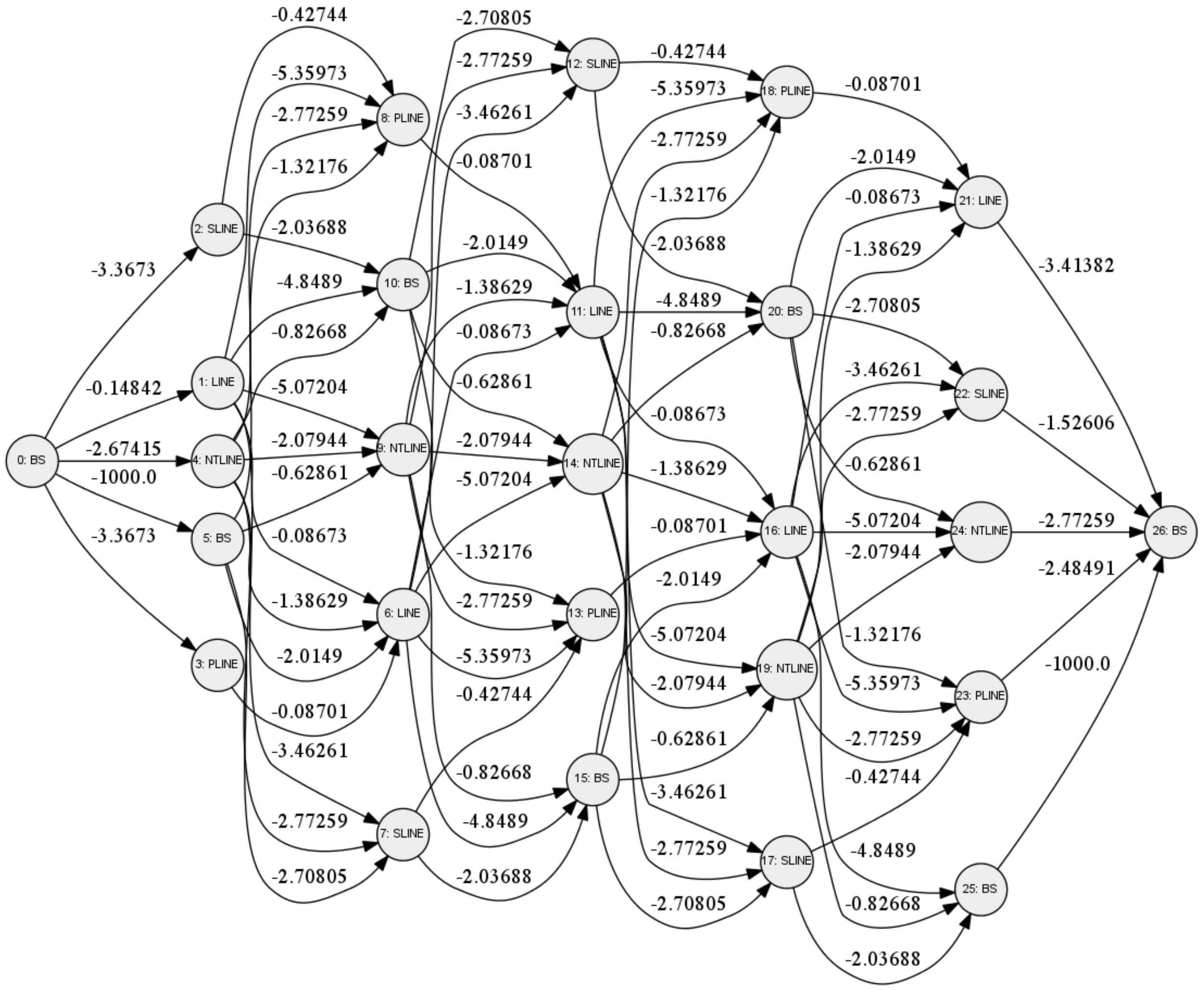

Figure 3.8: Example of a Line-constrained Language Model. Model created by replicating a Conditional Language Model 5 times.

3.3 Preprocessing Module

Our base scanned input images of the handwritten documents require that it's visual characteristics are improved/corrected in order to not impact adversely on the subsequent feature extraction and line detection processes. This process is done due to the known issues of handwritten historical documents:

- Low quality
- Stains and faint letters
- Loose formatting
- Narrow spaced lines
- Connected and Overlapping components
- Writing of the verso appearing on the recto due to bleed through

The following preprocessing techniques are applied:

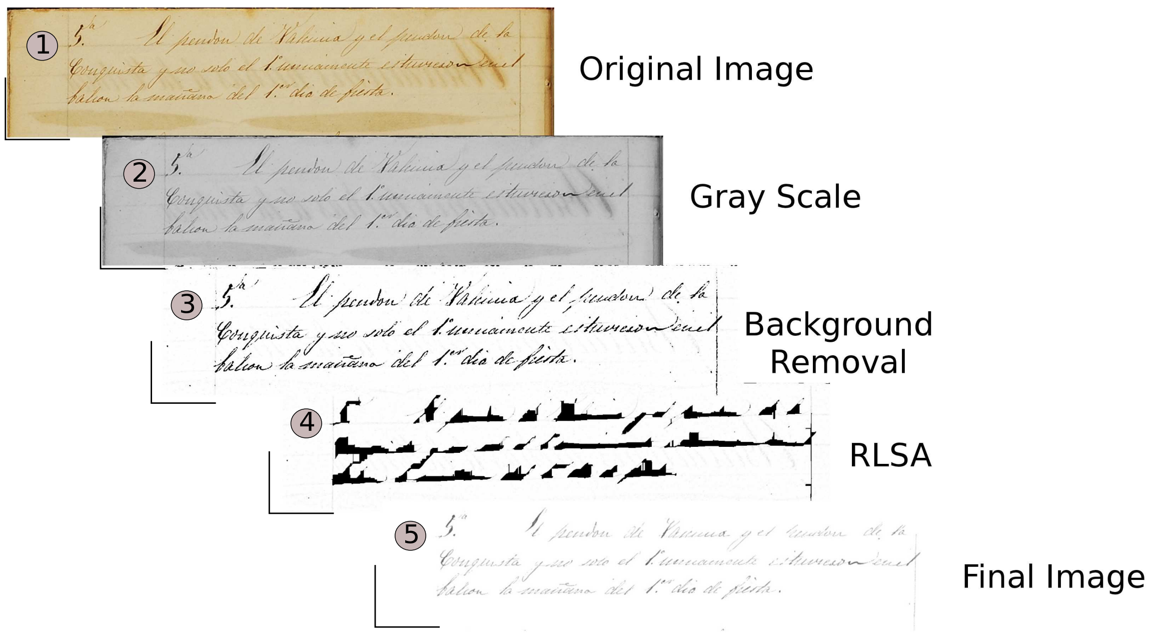

Figure 3.9: Figure shows the effects of each of the preprocessing phase subprocesses on a sample text region.

Background removal and noise reduction

We start from the original image (step 1 of Fig. 3.9) that contains stains, writing on the verso appearing on the recto and non uniformity of the background colour which makes it difficult to process. In order to eliminate these issues we first perform a grey-level normalization (step 2 Fig. 3.9) and apply on the resulting image a bi-dimensional median filter [3]; we subtract the result of the filter from the original image and obtain the result displayed in step 3 of Fig. 3.9.

Skew correction

Once the background is cleaned we can proceed to correct the skew. Skew is a distortion introduced during the document scanning process. It is understood as the angle of the document paper with respect to the scanner coordinates system. Skew must therefore be corrected one page image at a time, by aligning the text lines present with the horizontal axis.

Skew correction is carried out by searching for the angle which maximizes the variance of the vertical projection profile and then applying a rotation operation with the calculated angle [1]. We first execute the run length smear algorithm (RLSA) [6] to enhance the vertical projection profile (step 4 of Fig. 3.9) and then we calculate the angle which we use to correct the skew by applying a rotation operation. We obtain the final clean and skew corrected seen in step 5 of Fig. 3.9.

3.4 Feature Extraction

Since our TLAD approach is based on HMMs, each preprocessed image I (dimensions $M \times L$) must be represented as a sequence of feature vectors. This is done by dividing the already preprocessed image into D non-overlapping rectangular regions (from left-to-right) with height equal to the image-height L and calculate the projection profiles in each region (see Fig. 3.10).

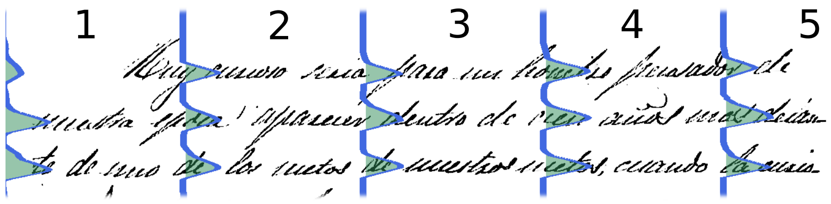

Figure 3.10: Partial page image visualization of 5 ($D = 5$) rectangular regions across over 3 handwritten text lines. For each region, its vertical projection profile is also plotted.

In oder to calculate the projection profile we must first binarize the image in order to differentiate foreground from background. We perform an Otsu binarization on the original image I obtaining the binarized image B.

For each of the pixel lines of the region $d : 1 \leq d \leq D$ of width m ,where $m = \frac{M}{l} D$ we compute the vertical projection profile value for an specific line $l : 1 \leq l \leq L$ on the image B:

$$o_{dl} = \frac{\sum\limits_{\substack{\forall j: \\ ((d-1)\cdot m)<j\leq(d\cdot m)}} B(i,j)}{\sum\limits_{\substack{\forall k: \\ 1\leq k\leq L}} \sum\limits_{\substack{\forall j: \\ ((d-1)\cdot m)<j\leq(d\cdot m)}} B(k,j)} \tag{3.6}$$

where all feature vectors $\overrightarrow{o_d}$ can be calculated with a time computational complexity of $\Theta(n + (L \cdot D))$, where n is the number of pixels in B .

With the calculated projection profiles, the D-dimensional feature vector is constructed for each page/block image row of pixels, by stacking the D projection profile values corresponding to that row. Hence, at the end of this process, a sequence of L D-dimensional feature vectors is obtained (see Fig. 3.2).

In order to improve the features extracted from the page image we used the RLSA algorithm to smear the lines prior to the vertical projection profile calculation to produce a more emphasized projection. Additionally, we decided to smooth the profile with the help of a rolling average filter [4] in order to eliminate noisy local maxima. Schematics of the resulting effects of the application of these methods can be seen in Fig. 3.11 and also in a real sample in Fig. 3.12

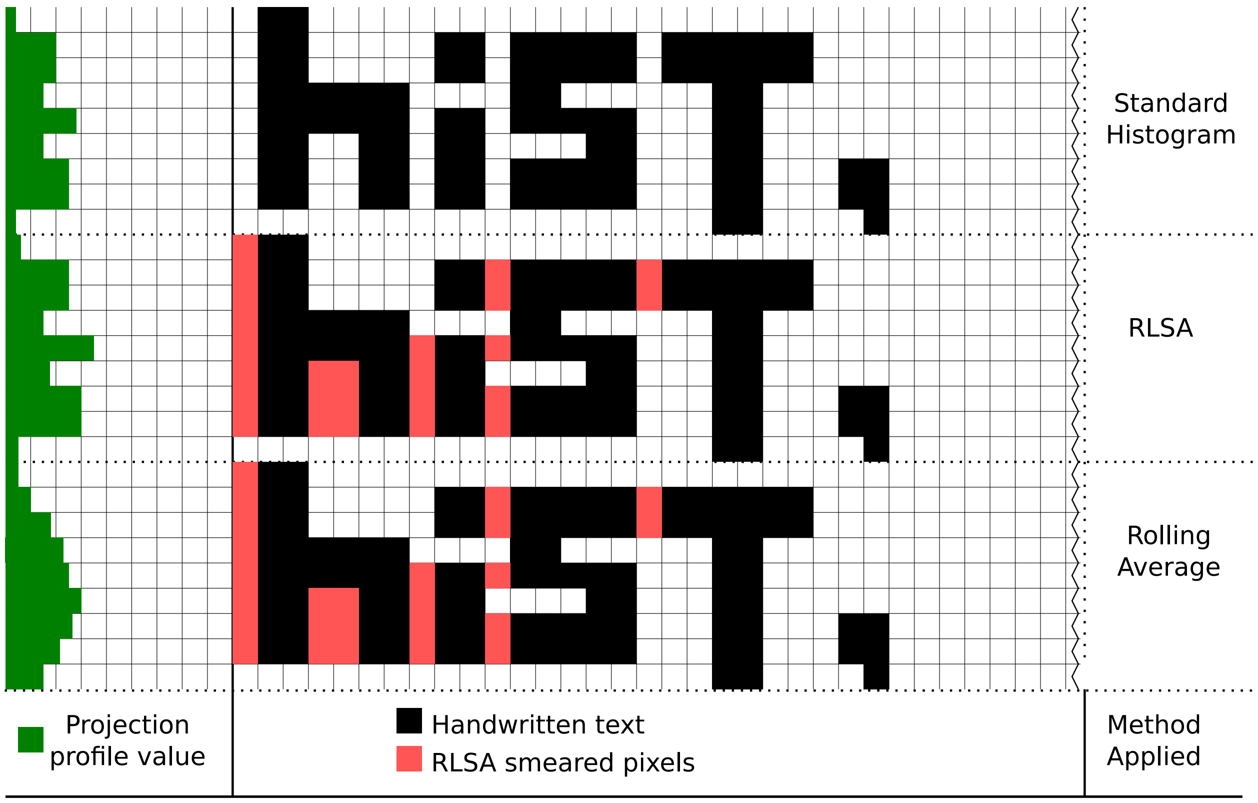

Figure 3.11: Schematics of the impact of the RLSA and rolling median filter on the vertical projection profile calculation.

3.5 Evaluation Measures

In order to assess the quality of the proposed TLAD approach, two kinds of measures have been adopted: "line error rate" (LER), considered a qualitative measure, which is calculated as the number of incorrectly assigned line labels divided by the total correct line regions; and the "alignment accuracy rate" (AAR) which, addressing the evaluation more from a quantitative point of view, measures the geometrical accuracy of the detected baseline coordinates in respect to the corresponding (correct) reference marks.

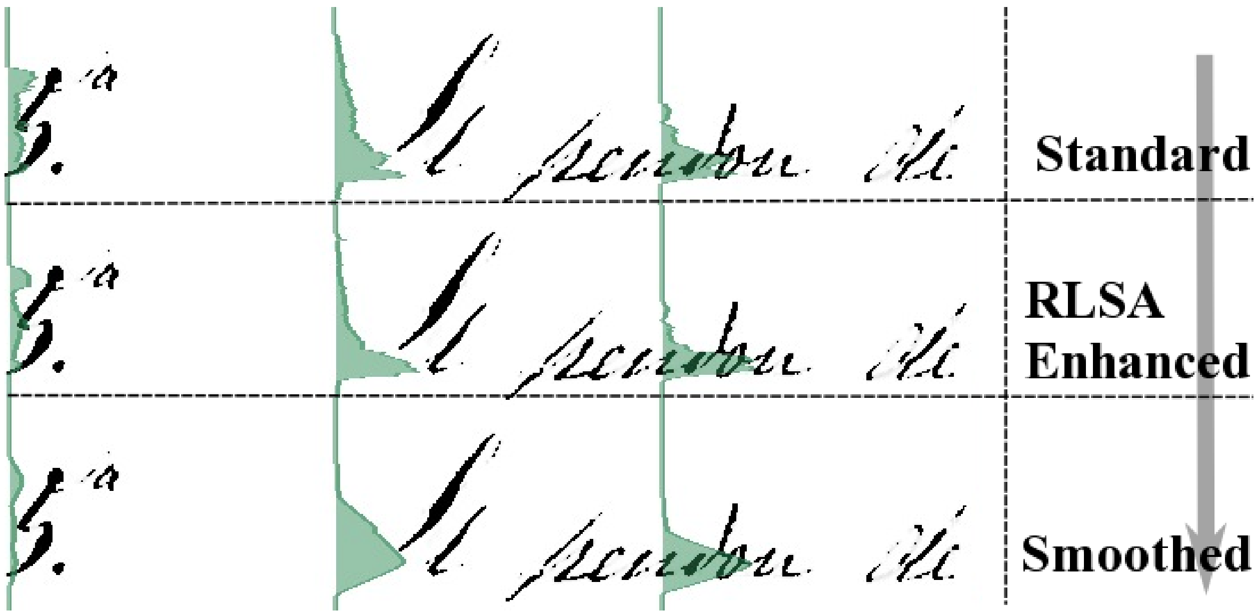

Figure 3.12: View of the impact of the RLSA and rolling median filter on the vertical projection profile calculation of a sample line.

The LER is obtained by comparing the sequences of automatically obtained region labels ($\hat{\mathbf{h}}$ in eq. 3.5) with the corresponding sequences. This is computed in the same way as the well known WER, with equal editing-costs assigned to deletions, insertions and substitutions [5].

LER is computed for two cases:

- Detection - All types of text lines are considered equal and we only measure the accuracy of the system in differentiating text lines from blank spaces.
- Classification - The full line type set indicated in subsec. 3.2.2 are maintained and we consider the miss-classification between the different text line types and the reference labels.

The AAR evaluation measure is calculated in three phases. First, for each page, we find the best alignment between the system-proposed baseline coordinates ($\hat{\mathbf{b}}$ in eq. 3.5) and the page reference baseline coordinates ($\mathbf{r}$) by minimizing the accumulated absolute difference. This minimization can be performed by means of dynamic programming.

We define P as a possible alignment (list of operations) between two list of baselines (b and r)i:

$$\hat{\mathbf{b}} = \langle \hat{b_1}, \hat{b_2}, \ldots, \hat{b_n} \rangle \tag{3.7}$$

$$\mathbf{r} = \langle r_1, r_2, \ldots, r_m \rangle \tag{3.8}$$

$$P = \{p_k = (\hat{b_{ik}}, r_{jk}) : \hat{b_{ik}} \in \hat{\mathbf{b}} \wedge r_{jk} \in \mathbf{r} \wedge 1 \leq k \leq n + m\} \tag{3.9}$$

We define the cost $c(k)$ of the alignment k as: $c(k) = |b_{ik} - r_{jk}| \cdot w_k$. Hence we can define $W(P)$ as the ponderated total cost of the sequence of operations in the following manner:

$$W(P) = \sum_{k=1}^{|P|} c(k) \tag{3.10}$$

where w_k is the specific ponderation weight associated to the alignment operation k, which takes the following values: 1 for insertion and deletion and 2 for substitution.

The alingment cost of two list of baselines $\mathbf{b}$ and $\mathbf{r}$ can be calculated by means of the unnormalized edit distance $d(\mathbf{b}, \mathbf{r})$:

$$d(\hat{\mathbf{b}}, \mathbf{r}) = \min_P \frac{W(P)}{L(P)} \tag{3.11}$$

where $L(P)$ is defined as the number of elementary ponderated edit operations described by P. As we assume the cost of substitution to be double the cost of insertion or deletion $L(P)$ is actually a constant K, for all possible alignment paths. Thus, we ensure that substitution is not favoured over insertion and deletion, therefore allowing us to simplify the last equation to:

$$d(\hat{\mathbf{b}}, \mathbf{r}) = \frac{1}{K} W(\hat{P}) \tag{3.12}$$

where $\hat{P} = \min_P W(P)$ which can be easily resolved by traditional dynamic programming technique.

Finally as we want the actual non-ponderated difference between the reference and hypothesis coordinates we define the real cost $d_r(\mathbf{b}, \mathbf{r})$ as the non ponderated sum of differences of the minimum cost alignment $\hat{P}$.

$$d_r(\mathbf{b}, \mathbf{r}) = \sum_{k=1}^{|\hat{P}|} |b_{ik} - r_{jk}| : (b_{ik}, r_{jk}) \in \hat{P} \tag{3.13}$$

A graphical representation of an alignment cost calculation can be seen in figure 3.13.

In the second phase in order to obtain a global measure, we calculate the mean value (and standard deviation) of the real cost per text line for the total number pages G. In order to do this we define $\mathbf{l_g}$ as the subsequence of text line reference labels of the corpus page g:

$$\mathbf{l_g} = \langle l_1, l_2, \ldots, l_s : \forall 1 \leq i \leq s : l_i \in \{\mathrm{NL}, \mathrm{PL}, \mathrm{SL}\} \rangle \tag{3.14}$$

with which we calculate our global measures:

$$\mu = \frac{\sum_{g=1}^{G} d_r(\mathbf{b}_g, \mathbf{r}_g)}{\sum_{g=1}^{G} |\mathbf{l_g}|} \tag{3.15}$$

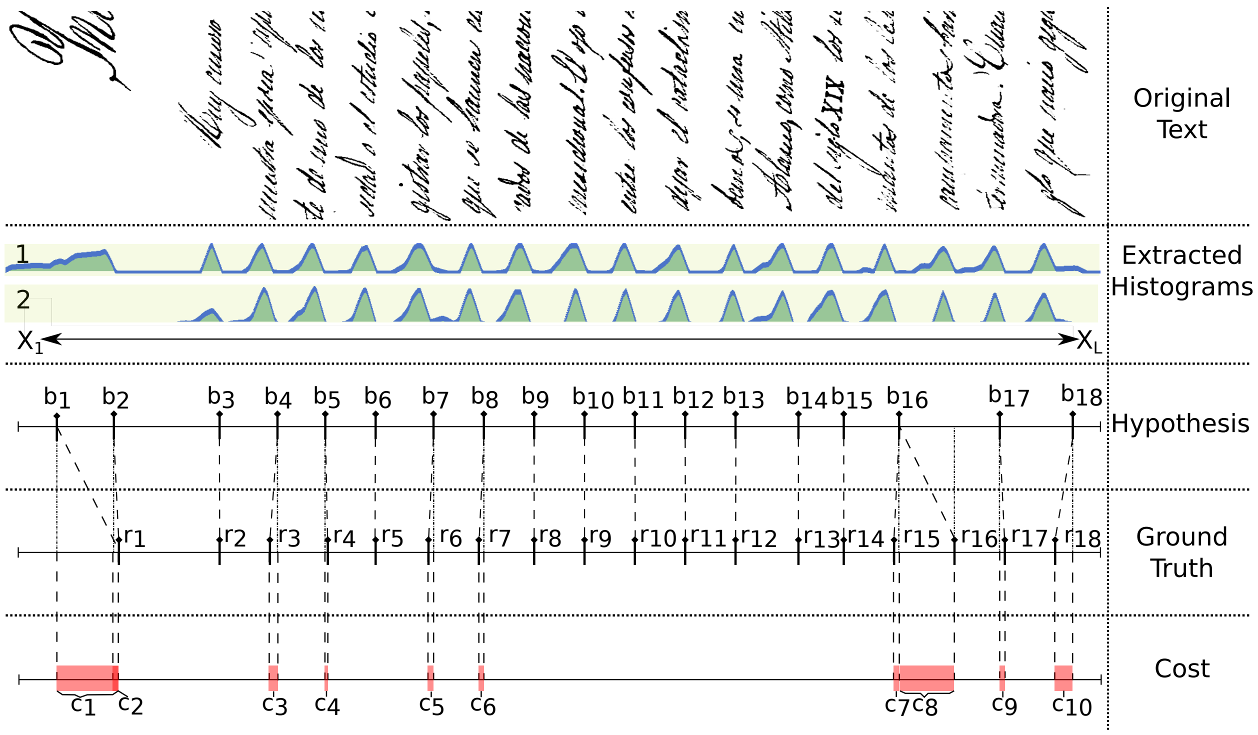
Original Text
Extracted Histograms
1
2
X1
XL
Hypothesis
b1 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11 b12 b13 b14 b15 b16 b17 b18
Ground Truth
r1 r2 r3 r4 r5 r6 r7 r8 r9 r10 r11 r12 r13 r14 r15 r16 r17 r18
Cost
c1 c2 c3 c4 c5 c6 c7 c8 c9 c10

Figure 3.13: Figure shows the schematics of the alignment cost calculation for a single page.

$$\sigma = \sqrt{\frac{\sum_{g=1}^{G}(d_r(\mathbf{b}_g, \mathbf{r}_g) - \mu)^2}{\sum_{g=1}^{G}|\mathbf{l_g}|}} \tag{3.16}$$

The resulting mean and standard deviation of the second phase measure the deviation of in pixels. In order to make this measure independent of the page resolution we present it as a percentage of the average height of a text line (in pixels) h:

$$\mu_r = \frac{\mu}{h} \times 100 \tag{3.17}$$

$$\sigma_r = \frac{\sigma}{h} \times 100 \tag{3.18}$$

3.6 Conclusions

During the course of this past chapter we have:

- Provided an in depth description of our detection and classification system
- Detailed the procedure to preprocess the pages and lines to enhance the performance
- Described the feature vector extraction process
- Defined the required modelling to represent the types of vertical regions and how they are composed to define a page
- Presented two types of evaluation measures required to evaluate the performance of our system

Bibliography

[1] i Gadea, M. P. (2007). *Aportaciones al reconocimiento automático de texto manuscrito*. PhD thesis, Universidad Politécnica de Valencia. Advisors: Enrique Vidal and Alejandro H. Toselli.

[2] Jelinek, F. (1998). *Statistical Methods for Speech Recognition*. MIT Press.

[3] Kavallieratou, E. and Stamatatos, E. (2006). Improving the quality of degraded document images. In *Document Image Analysis for Libraries, 2006. DIAL '06. Second International Conference on*, pages 10 pp. –349.

[4] Manmatha, R. and Srimal, N. (1999). Scale space technique for word segmentation in handwritten documents. In *Proceedings of the Second International Conference on Scale-Space Theories in Computer Vision*, SCALE-SPACE '99, pages 22–33, London, UK. Springer-Verlag.

[5] McCowan, I. A., Moore, D., Dines, J., Gatica-Perez, D., Flynn, M., Wellner, P., and Bourlard, H. (2004). On the use of information retrieval measures for speech recognition evaluation. Idiap-RR Idiap-RR-73-2004, IDIAP, Martigny, Switzerland.

[6] Wong, K. Y., Casey, R. G., and Wahl, F. M. (1982). Document Analysis System. *IBM J.Res.Devel*, 26(6):647–656.

CHAPTER 4

EXPERIMENTS AND RESULTS

Chapter Outline

4.1 Experimental Set-up and Baseline

In order to review the performance and quality of our line detection and classification system we will perform a series of experiments on the "Cristo-Salvador" (CS) corpus. All experiments will be carried out using hold-out as per the CS hard partition, described in detail in section 4.2.

For the detection task considered all vertical text line region types (NL, SL and PL) as a unique class while in the classification task we consider the full set of types.

To summarize we will review the following major experiment set-ups:

- Review the impact of the Language Model on detection and classification accuracy. To perform this we will perform experiments with the prior, conditional and line number constrained language models.
    - Prior and conditional models are generated by maximum likelihood from the training set.
    - Line number constrained models are generated for each of the test pages. They are built considering the number of vertical regions present in the page and use the probabilities of the conditional model to populate the transitions.
- Study the impact of hard constrained morphological models on the accuracy. In order to review this we have performed experiments with a fix number of states with strictly linear models and also ranged linear models where the minimum and maximum number of states to consider are computed from the training set.

For all the above major experiment set-ups we have tuned to the following parameters:

Number of columns: The number of vertical regions D to consider. For each of the regions we will calculate the grey histogram level for each pixel line.

Column overlap: Percentage of overlapping allowed between regions.

Training iterations: Number of cycles used to train the different morphological models.

Number of Gaussians: Number of Gaussians to consider for the Gaussian mixture that governs the emission probability at each of the HMMs states.

Grammar scale factor (GSF): factor by which the language model likelihoods must be post multiplied. This factor is used in order to increase/reduce the impact of language model on the decoding process.

Word insertion penalty (WIP): word insertion log probability to consider while decoding test samples. The WIP is used to control the number of words that are considered while decoding.

For comparison purposes, we obtained a baseline result by employing a standard, line detection approach based on plain whole-line vertical projection profiles [6]. The method requires as input the expected number of text lines to be found on the corresponding page and yields the baseline coordinates without any labelling information.

4.2 Corpora Description

Experiments were carried out using a corpus compiled from a XIX century Spanish manuscript identified as “Cristo-Salvador” (CS), which was kindly provided by the *Biblioteca Valenciana Digital* (BiVaLDi)[a]. This is a rather small document composed of 53 color images of text pages, scanned at 300 dpi and written by a single writer. Some page images examples are shown in Fig. 4.1.

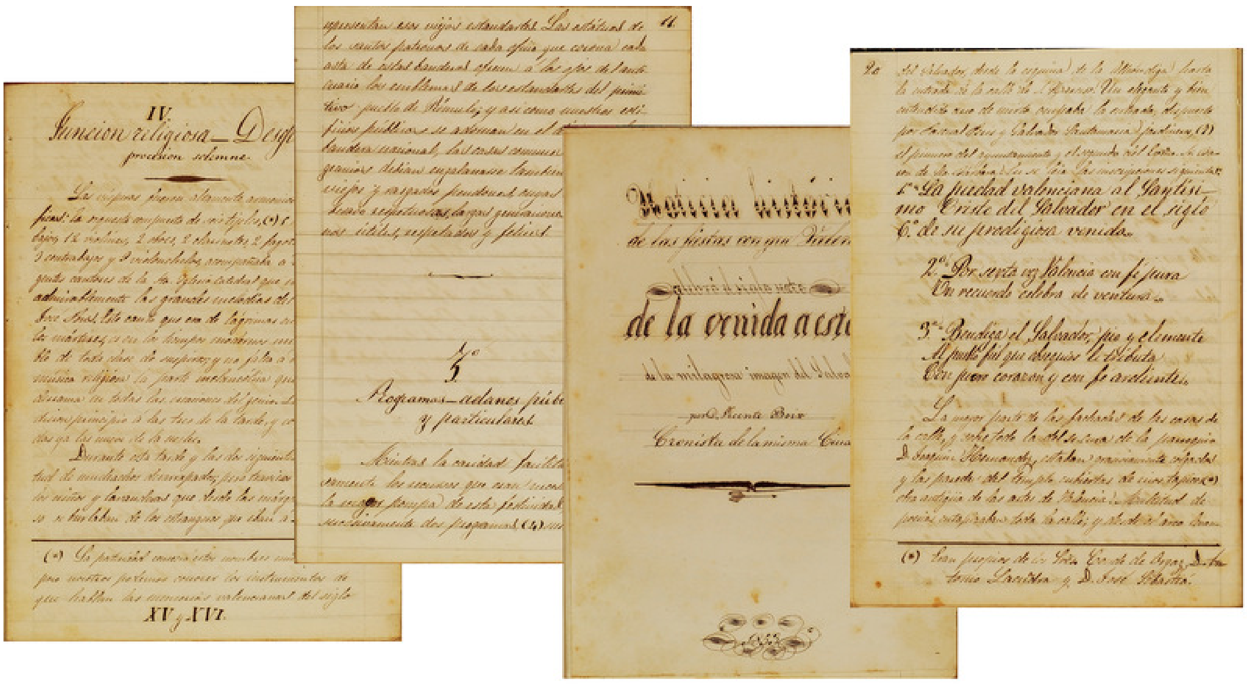

Figure 4.1: Examples of page images from CS corpus.

In this case. we employ the already predefined *book* partition [1], for which someexamples are shown in Fig. 4.1. This partition divides the data-set into a test set containing 20 page images, and a training set composed of the 33 remaining pages. Table 4.1 presents basic statistical information of the *book* partition.

Each page was annotated with a succession of reference labels (NL, SL, PL, NT, BL and IL) indicating the kind of line-regions it is composed of. Line positions were obtained in a first instance by executing standard methods for text line detection based on the whole-line vertical projection profile, which were afterwards manually labelled, verified, adjusted and/or rectified by a human operator to ensure correctness.

[a]http://bv2.gva.es.

Table 4.1: Basic statistics of the Cristo-Salvador corpus “book” partition.

Number of:	Training	Test	Total
Pages	33	20	53
Total text lines	685	497	1 182
Normal-text lines (NL)	638	442	1 080
Short-text lines (SL)	23	35	58
Paragraph-text lines (PL)	24	20	44
Blank Lines (BL)	73	70	143
Non-text Lines (NT)	16	8	24
Inter Lines (IL)	701	505	1 206

4.3 Results

4.3.1 Detection Experimentation

We initially start by reviewing the detection performance of our system with the Prior Model Language. We proceed by setting first the topology of the HMMs for the various Vertical Region Types. In order to do this we compare the detection LER when we vary the number of states in the HMMs and the number of Gaussians in each state:

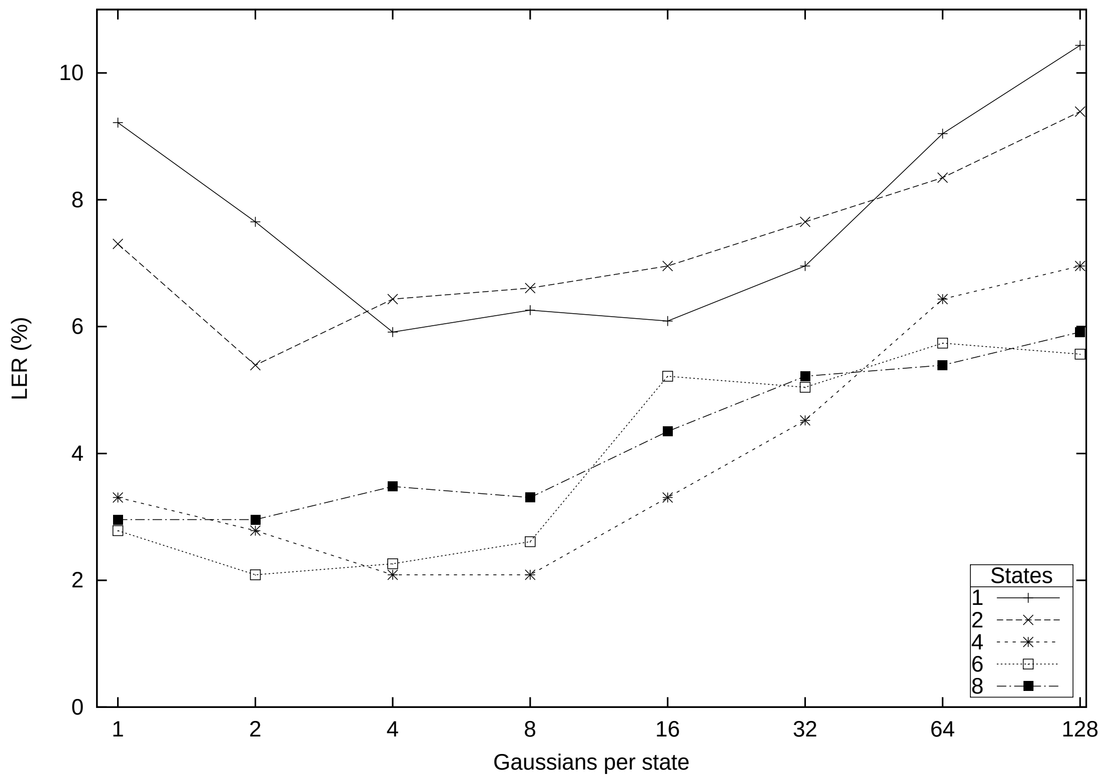

Figure 4.2: Plot shows the LER (%) for different HMMs number of states as a function of the number of Gaussians per state. Plot is performed for an specific number of columns extracted (8), overlap (20%) WIP (-32) and GSF (1).

In Fig. 4.2 we can see that:

- HMMs with a small number of states (1,2) provide an overall worse performance.
- HMMs with more states (4,6,8) provide better results. The topology with 4 states is better as there is no significant difference with the other topologies (6,8) but requires less parameters.
- Independently of the number of states the performance degrades for all topologies when we increase the number of Gaussians per state over the value 8. This is expected as an increase in the number of Gaussians implies an increase in the number of parameters to train while the number of training samples remain the same.

As per the above comments we select as best topology 4 states with 8 Gaussians per state. Once the Topology is set we evaluate the number of horizontal vertical regions to divide the image pages with, extracted columns, and the overlap between the regions.

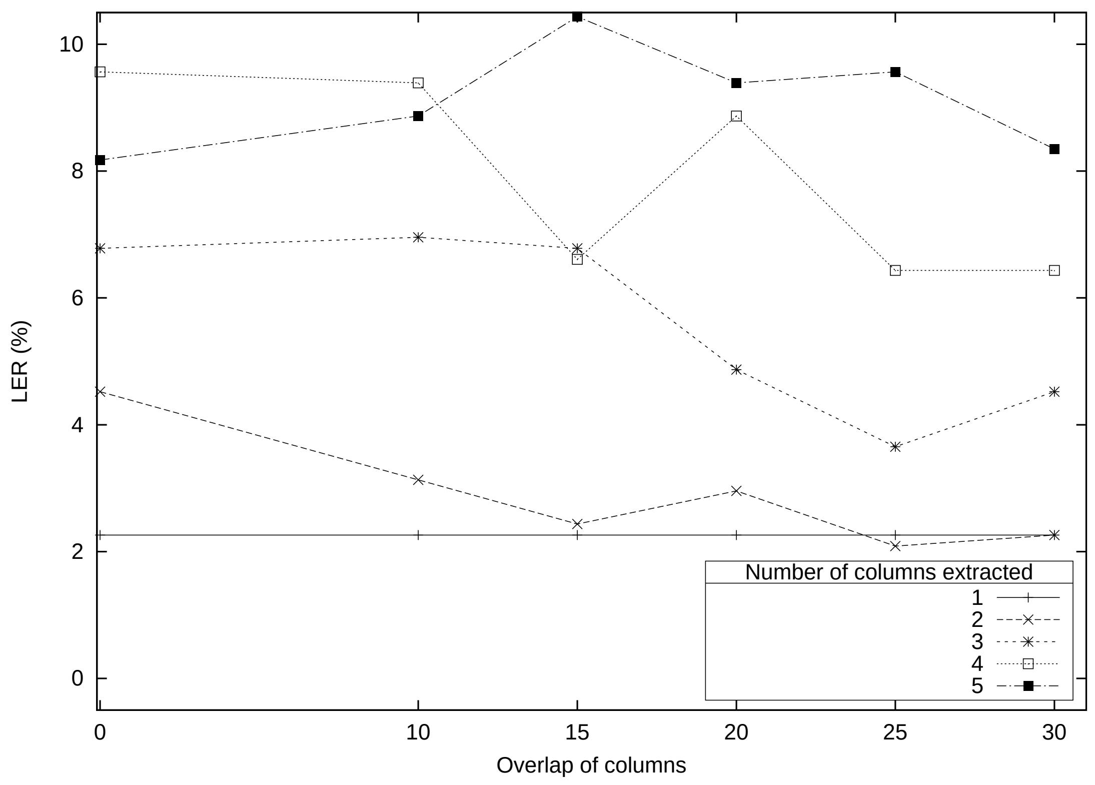

Figure 4.3: Plot shows the LER(%) for different number of columns extracted as a function of the region overlap percentage. Plot is performed for an specific number of HMM-states (4), Gaussians per state (8), WIP (-32) and GSF (1).

As we can see in Fig. 4.3 the best results for line detection are obtained when considering a low number of horizontal regions for feature extraction (one or two). This is expected as we have simplified the vertical region types: with a small number of features we can easily differentiate a text line from a blank space or non textual region.

Regarding the overlap within regions we can see that it impacts the end accuracy and we consider 25% as the best value. Next we will review the impact of the GSF and WIP.

It is shown in Fig. 4.4 and Fig. 4.5 that the smaller the GSF value is the more sparse the results are. The best possible value is reached in the zone of 8 to 16 GSF and afterwards it increases. Regarding WIP we can see that with greater insertion penalties we get better detection performance up to -32.

We have obtained the best result for the prior LM with the following configuration: HMM states (4), Gaussians per state (8), columns extracted (2), region overlap (25%), GSF (16) and WIP (-32). We perform the same process for the conditional LM and the line-number constrained LM.

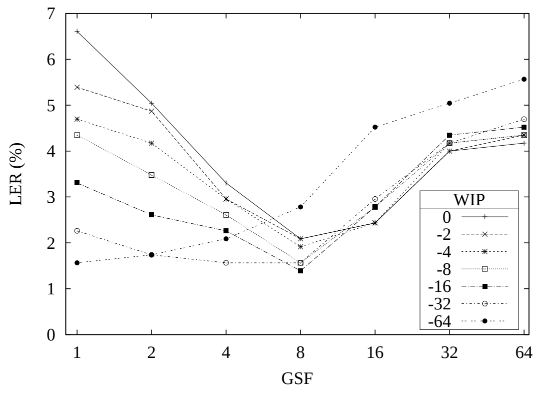

Figure 4.4: Plot shows the LER (%) for different number of WIP values as a function of the GSF value. Plot is performed for an specific number of HMM-states (4), Gaussians per state (8), columns extracted (1) and region overlap (25%)

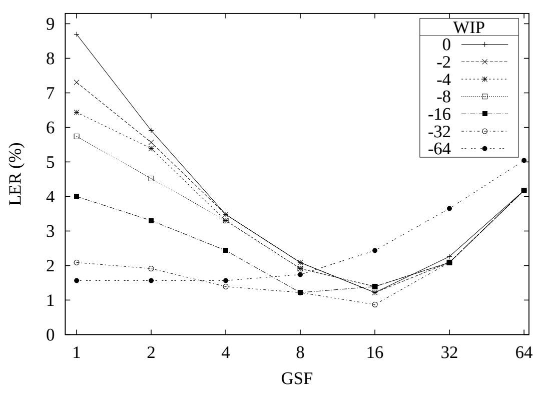

Figure 4.5: Plot shows the LER (%) for different number of WIP values as a function of the GSF value. Plot is performed for an specific number of HMM-states (4), Gaussians per state (8), columns extracted (2) and region overlap (25%)

In Table 4.2 we report the best figures for LER and AAR achieved through the indicated experimentation process for the three LMs and the heuristic baseline method. The AAR mean and std-dev are given in this case in percentage of the text line average width (80 pixels).

Table 4.2: Best detection figures of LER(%) and AAR(%) obtained for our statistical text line analysis approach (STLAD) and the heuristic one (HEUR), using different kind of language models: Prior (PRI), Conditional (CND) and Line-Number Constrained (LN-C).

Approach	LM	LER(%)	AAR(%)	
			μ_r	σ_r
STLAD	PRI	0.86	9.04	15.91
	CND	0.70	8.81	15.40
	LN-C	0.34	8.75	13.15
HEUR	LN-C	–	9.94	29.84

Although the HEUR method does not formally use a LM it requires as input the number of text lines (NL) present in the page thus for comparison reasons we consider it to be using a LN-C model.

We observe a trend in Table 4.2: the more restrictive the LM is, the better accuracy is achieved. Similarly the quantitative evaluation shows that more construed LMs provide better baseline coordinate hypotheses (closer to the ground truth ones). In the case of the HEUR method, the obtained AAR (std) is not as good as the STLAD's with a much higher std-dev.

In image 4.6 we can also see the quantitative difference through visual comparison of our proposed method and the base projection method. Is this intuitive visualization we can observe that our method provides a frontier much closer to the bulk of the text thus effectively detecting better the baseline than the histogram projection method.

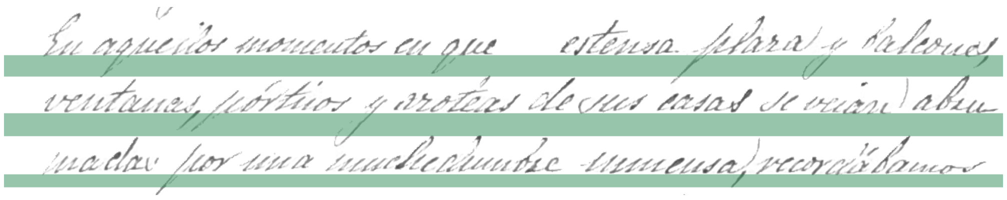

Figure 4.6: Image shows the difference between our proposed method (upper side of each coloured region) and the histogram projection method (lower side)

4.3.2 Classification Experimentation

For the classification experimentation the same process as for detection was applied. There are some specific aspects of the classification task that we will now illustrate.

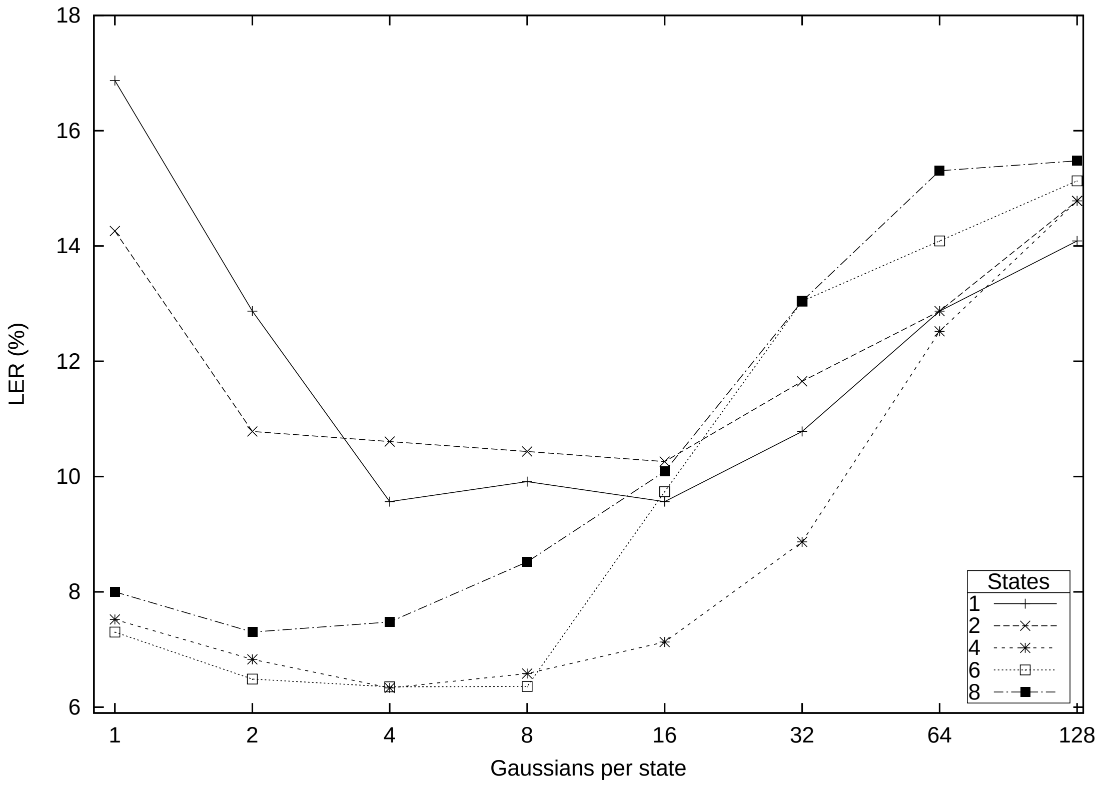

Figure 4.7: Plot shows the LER (%) for different HMMs number of states as a function of the number of Gaussians per state. Plot is performed for an specific number of columns extracted (8), overlap (20%) WIP (-32), GSF (1) with the Prior LM.

In Fig. 4.7 we note one of the main differences between the classification and detection tasks: the optimal number of Gaussians per state is reduced from 8 to 4. This reduction is due to the fact that we have added more classes thus also reducing the amount of training data for the text line vertical regions classes.

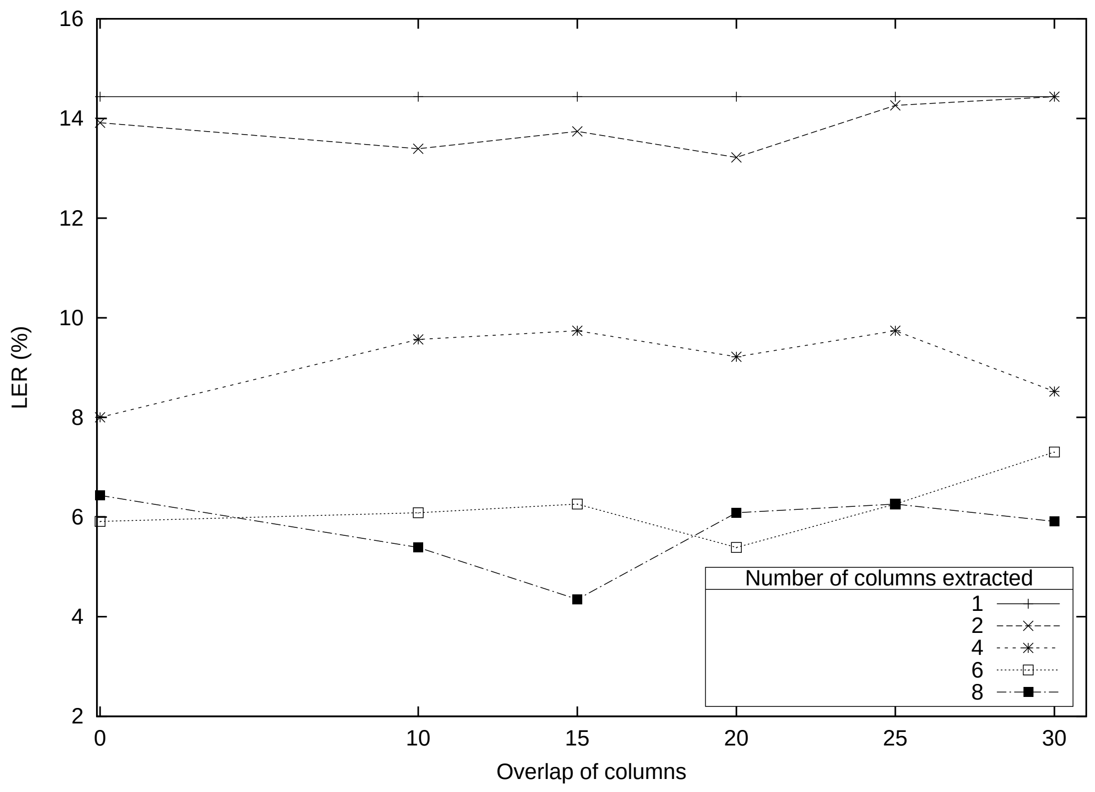

Figure 4.8: Plot shows the LER (%) for different number of columns extracted as a function of overlap percentage between the regions. Plot is performed for an specific number of states in HMM (4), Gaussians per state (4), WIP (-128), GSF (32) with the Prior LM.

In Fig. 4.8 we observe that when we evaluate text line classification better results are obtained when we consider a higher amount of page horizontal regions for the feature vector extraction in comparison to the detection task. This is logical as the only way to differentiate between the different text line types is to consider features which allow us to discriminate through the length or indentation of the lines.

We show in Table 4.3 the best figures for LER and ARR achieved for the classification task by the three LMs and the heuristic baseline.

We can observe in Table 4.3 that the LER values are higher than the ones of the detection task. This is mainly due to the increase in vertical region types, for which the amount of training samples is smaller due to the redistribution. In the classification task we can also see the positive effect of applying more restrictive LMs as they make a positive impact in the overall system accuracy.

Table 4.3: Best classification figures of LER(%) and AAR(%) obtained for our statistical text line analysis approach (STLAD) and the heuristic one (HEUR), using different kind of language models: Prior (PRI), Conditional (CND) and Line-Number Constrained (LN-C).

Approach	LM	LER(%)	AAR(%)	
			μ_r	σ_r
STLAD	PRI	6.44	9.22	27.71
	CND	4.7	8.92	23.25
	LN-C	4.2	8.88	20.25
HEUR	LN-C	–	9.94	29.84

The AAR measure is also impacted adversely by the reduction of training samples, specially in std-dev, but the system still outperforms the HEUR method.

4.3.3 Vertical Region Models Experimentation

Several test where performed with the best configurations of the LMs where the HMMs where changed as to use a linear ranged topology.

The range of the topology was learnt from the training data and several values where tried for:

- Minimum range:
  - Mean length of a text line
  - Mean length of a text line minus standard deviation
  - First percentile
- Maximum range:
  - Mean length of a text line
  - Mean length of a text line plus standard deviation
  - Third percentile

All combinations of both values where tested and the results provided did not provide any significant variation from the original results (with out linear ranged HMMs) for the same training and decoding parameters.

Although restricting the HMM topology did not provide an improvement on the classification and detection accuracy for this specific task the technique does seem to be promising and we expect it to have a positive impact in more complex corpus.

Bibliography

[1] Romero, V., Toselli, A. H., Rodríguez, L., and Vidal, E. (2007). Computer Assisted Transcription for Ancient Text Images. In *International Conference on Image Analysis and Recognition (ICIAR 2007)*, volume 4633 of *LNCS*, pages 1182–1193. Springer-Verlag, Montreal (Canada).

CHAPTER 5

GENERAL CONCLUSIONS AND FUTURE WORK

Chapter Outline

5.1 Conclusions

We have shown a new way of addressing text line analysis and detection by using a statistical framework, similar to the already employed in many popular ASR and HTR tasks, that avoids the traditional heuristics approaches generally used to solve this problem.

In comparison to the currently more widely used projection approach:

- Our approach requires a training phase and supervised data thus it is mostly suitable for large volumes with consistent page structure.
- Detection and classification with the new approach is performed in polynomial time thus being up to par in this aspect with the heuristic approaches.
- Not only does our method not require the input of the number of lines to detect in the page to work adequately, but also, through the language model, it provides us an easy way to introduce any structural information we may know.
- The proposed approach not only detects the baselines but is able to label the text lines; In this aspect surpassing current approaches.
- Our system yields baseline coordinates of better quality than the heuristic method.

5.2 Publications

The work presented in this paper has been submitted and accepted in:

- The sixth workshop on Language Technology for Cultural Heritage, Social Sciences and Humanities (LaTeCH) held in Avignon April 24th, 2012 [1]. LaTeCH is an international workshop associated to the European Chapter of the Association for Computational Linguistics.
- The thirteenth International Conference on Frontiers in Handwriting Recognition (ICFHR) that will be held in Bari on September 18-24, 2012.

5.3 Future Work

Even though a considerable amount of time and work has gone into the realization of this research, there are still many aspects to explore. The following extensions could be performed:

Explore other options for line detection: Currently our approach requires that the images passed contain roughly horizontal text lines in order work adequately. This assumption causes our approach to not be feasible for some historical documents and also for free form text a user might write in an notepad.

Conduct more experiments on other corpora: Our approach has only been tested on the "Cristo-Salvador" corpus. It would be interesting to use others to verify that our obtained data/results are reliable. It is envisioned that the proposed stochastic framework serves as a cornerstone to implementing interactive approaches to line detection similar to those used for handwritten text transcription used in [3]. We envision to provide an e-pen interface to allow the user to correct the initial output. The user would be able to perform the following actions:

- Perform a gesture to indicate that the current assigned vertical region label is incorrect at which point the system would provide a list of other possible labels found.
- Correct the line detected by adding mandatory pass way-point through the e-pen which would force the line path to be recalculated to accommodate for it.
- Signal the system, by selecting an area with the e-pen, where regions have not been identified forcing the refine the detection process of that zone.
- Cross out detected regions that are in reality non existing in the page.
- In the event of having connected components of different lines that have been wrongly segmented or assigned to a line the user could correct this through a gesture.

Use Adaptive Learning to improve the recognition through user's feedback: As the user corrects or validates the vertical regions labels and the baseline coordinates the system can use this new information to further train the statistical model or adapt it to the current task in order to the systems accuracy. In speech recognition, well known Adaptive Learning techniques exist for adapting the acoustic HMM models to the speaker [4] [2] which could be use for our intended purpose.

Bibliography

[1] Bosch, V., Toselli, A. H., and Vidal, E. (2012). Natural language inspired approach for handwritten text line detection in legacy documents. pages 107–111.

[2] Pitz, M., Molau, S., Schlüter, R., and Ney, H. (2001). Vocal tract normalization equals linear transformation in cepstral space. In *IN PROC. OF THE EUROSPEECH 01*, pages 2653–2656.

[3] Toselli, A. H., Romero, V., Pastor, M., and Vidal, E. (2009). Multimodal interactive transcription of text images. *Pattern Recognition*, 43(5):1824–1825.

[4] Woodland, P. C. (2001). Speaker adaptation for continuous density HMMs: A review. In *ITRW on Adaptation Methods for Speech Recognition*, pages 11–19.