Diplomarbeit im Fach Informatik

Improved Modeling in Handwriting Recognition

Der Fakultät für
Mathematik, Informatik und Naturwissenschaften der
RHEINISCH-WESTFÄLISCHEN TECHNISCHEN HOCHSCHULE AACHEN

Lehrstuhl für Informatik VI
Prof. Dr.-Ing. H. Ney

vorgelegt von:
Stephan Jonas
Matrikelnummer 249175

Gutachter:
Prof. Dr.-Ing. H. Ney
Prof. Dr. B. Leibe

Betreuer:
Dipl.-Inform. Philippe Dreuw

Juni 2009

Erklärung

Hiermit versichere ich, dass ich die vorliegende Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Alle Textauszüge und Grafiken, die sinngemäß oder wörtlich aus veröffentlichten Schriften entnommen wurden, sind durch Referenzen gekennzeichnet.

Aachen, im Juni 2009

Stephan Jonas

Abstract

In this work a script independent handwriting recognition system is proposed which is derived from the RWTH-ASR hidden Markov model (HMM) based speech recognizer.

Most problems occurring in handwriting recognition (HWR) are induced by large variations within the written text. In particular, different handwriting styles such as cursive writing or long drawn-out strokes are difficult to model.

Common handwriting recognition systems use various preprocessing and feature extraction methods to compensate for the variations in handwriting. Additional approaches have been made using writer adaptive training for writer dependent modeling of the variations.

This work uses another approach by dealing with these variations using explicit background modeling, improving the visual models of the handwriting and exploiting the character context within the script. Therefore, only simple appearance based features and only few preprocessing steps have to be used. Instead, methods known from handwriting recognition, such as model length estimation and discriminative training for writer independent modeling as well as common methods from speech recognition are applied. In addition, several approaches for improvement of continuous text line recognition are made by using lexica and language models accounting for the specialties in line recognition.

The script independence of the proposed system is demonstrated on Arabic and Latin recognition tasks. The performance is evaluated on the IFN/ENIT database which provides an Arabic single word recognition task. In addition, the IAM database is chosen, providing a Latin text line recognition task. The results obtained on both databases outperform the currently known best error rates achieved with a single recognition system.

Acknowledgment

I would like to express my gratitude to all people who have supported me during the process of this work:

Prof. Dr.-Ing. Hermann Ney for introducing me to pattern recognition, offering me the opportunity to work in the field of image recognition at the Chair of Computer Science 6 and for the possibility to write my diploma thesis at this department,

Prof. Dr. Bastian Leibe who kindly accepted to co-supervise this work on short notice,

Philippe Dreuw for his supervision of this work, many ideas, productive conversations, hints on bugs and not always being serious,

Thomas Deselaers for his dedication and serving as an unintentional mentor,

the image recognition group, especially Jens Forster for proof reading, suggestions and explanations on various topics, and Christian Oberdörfer and Patrick Röder for being irritated by the same bugs,

Verónica Romero Gómez and Nicolás Serrano Martínez-Santos from the Universidad Politécnica de Valencia for their preprocessing software, data and ideas on the IAM database,

Arne Mauser for help on language models and letting me use the cluster during thread level orange, David Rybach, BJörn Hoffmeister, Christian Gollan, Georg Heigold, Stefan Hahn and Saša Hasan for help on software and databases,

the “Hiwiraum” for all the fun it provides and the people sharing it with me,

Andreas Hannig for last minute proof reading and distracting me every now and then,

my fiancé Svenja for proof reading even without understanding and being there all the time, most of it listening patiently, and of cource special thanks to my parents for their support in every way during the last years, weeks and days.

Contents

Chapter 1

Introduction

HWR - as part of the optical character recognition (OCR) - is a relatively new field of computer vision. Modern OCR had its beginnings in the year 1951 with the invention of GISMO, an optical reader which was able to read typewritten text, Morse and musical notes by David H. Shepard. Two years later, he registered his “apparatus for reading” as US-patent (see [Shepard 53]). HWR first appeared in the late 1960s with the recognition of zip codes on letters or account and amount information on checks. The computer capacity of those days was not sufficient for large scale HWR. Even the recognition of printed text was only adequate on simplified fonts, for example the font used on credit cards ever since.

According to [Verma & Blumenstein$^+$ 98], today’s OCR systems are capable of recognizing typewritings in several fonts. Yet the offline HWR itself is an open topic in research, whereas online HWR is commonly used in PDAs, tabletop PCs and even game consoles. Online HWR uses a sorted stream of positions of the pen during the writing (see [Tappert & Suen$^+$ 90, Plamondon & Srihari 00, Koerich & Sabourin$^+$ 03]), which offers more informations than the image of the written text used in offline HWR. Corresponding to [Bunke & Roth$^+$ 95, Bertolami & Bunke 08], high recognition accuracy is still difficult to achieve in offline continuous HWR. The reason for this development has to be seen in the high variability such as individual writing styles, the number of different word classes and the word segmentation problem, which comes with handwriting.

On the other hand the need fore a large scale recognition system allowing for continuous HWR is growing, since most texts and information nowadays are required to be in digital form rather than paper based. Nonetheless, handwriting is still one of the main approaches for people to collect and store data. Additionally, a huge amount of text is only available in handwriting, for example, almost all historical documents or all information gathered by forms (e.g. in surveys, administrative forms, etc).

Hence, the aim of this work is the development and evaluation of a script and writer independent HWR system using the HMM based speech recognizer RWTH-ASR (see [Rybach & Gollan$^+$ 09, Lööf & Gollan$^+$ 07]). In difference to most state of the art HWR systems, which devote a lot of effort on the preprocessing and feature extraction (cf. Chapter 2), only little preprocessing and simple appearance based features are

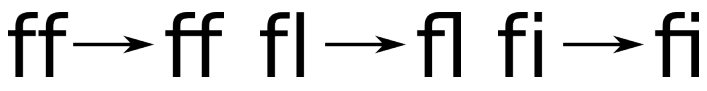
used. The main focus of the thesis is set to context-, character-, word- and visual modeling. The impact of methods derived from current literature in HWR and automatic speech recognition will be evaluated and combined. For context modeling, classification and regression tree (CART) is used which is well known from speech recognition (see [Beulen & Bransch$^+$ 97]). Character and word models are improved using model length estimation (MLE) and white space modeling which have been published recently by [Dreuw & Jonas$^+$ 08, Schambach 03]. Various authors have identified the creation of statistical language models (LMs) to be a very important step in continuous HWR (e.g., see [Bunke & Bengio$^+$ 04, Pitrelly & Roy 03, Marti & Bunke 02b, Plamondon & Srihari 00]). Therefore, this matter is also discussed in details in this work.

In Chapter 1 a short overview of HWR and the handwriting systems used for evaluation of the proposed methods is given. The remainder of this work is organized as follows: Previous work and current achievements in HWR are presented in Chapter 2. In Chapter 3 the theoretical background of the used system and the proposed methods for preprocessing, feature extraction and modeling are described. The databases and corpora used for the evaluation of the methods are listed in Chapter 4. The results obtained are discussed in Chapter 5. Finally, a summarization and conclusion of this work is given in Chapter 6.

This thesis deals with the recognition of Latin and Arabic handwriting. Since the handwritings for both languages differ substantially and accordingly require different approaches of HWR, a short overview of both languages and notations will be given.

1.1 Latin Handwriting

The Latin handwriting is used for Western languages and is one of the most common handwritings worldwide. It makes use of the modern Latin alphabet, the so-called "Arabic" or "Indian" digits, and letterpress punctuation. English handwriting, which will be used in this work, uses the alphabet's 26 basic characters and does not make use of additional symbols like other European languages. Each letter can be written in lower- and uppercase. Additionally, two styles are used, the block or capitalized writing and the cursive writing. Therefore, four writing variants are known for each letter. The concatenation of at last two letters, so-called "ligatures", are theoretically possible but seldom used in handwriting. For example, the character "&" is historically a ligature. Other ligatures, more commonly used in typewriting are displayed in Figure 1.1. For more details on ligatures see [Neubauer 96].



(a) ff ligature (b) fl ligature (c) fi ligature

Figure 1.1: Example ligatures used in Latin typewriting

1.2 Arabic Handwriting

Arabic handwriting differs from Latin handwriting in several ways. In contrast to English handwriting, Arabic is written from right-to-left and does not distinct between upper- or lowercase characters. 28 basic characters with 18 different basic shapes are used, but the appearance of a character can vary depending on its position in a word. According to [Majidi 06] four positions are possible for most characters (for example images see Table 1.1):

Isolated (or Alone)
: The character is not connected on either side.

Initial (or Beginning)
: The character starts a contiguous group of characters and is connected to the other characters from the left.

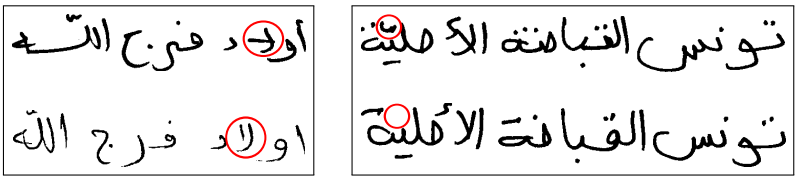
Medial (or Middle)
: The character stands within a group of contiguous characters and is connected to other characters on both sides.

Final (or End)
: The character is the last character of a contiguous group of characters and is thus connected from the right side.

Excepted from this circumstance are six characters which cannot be connected from the left side. Hence, a single Arabic word does not have to be a completely connected string of characters. In contrary, most words consist of several unconnected “pieces of Arabic word” (PAWs).

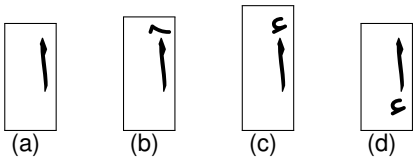
Similar to Latin handwriting it is possible to contract two or three defined characters into a ligature (e.g., see Figure 1.2(a)). In Arabic, ligatures are more commonly used than in Latin handwriting and they are handled as single characters. Thereby, their appearance can also vary depending on their position within the word. Again, some exceptions exist depending on the characters that are contracted.

Different from these normal ligatures are so-called “Chadda ligatures”. The Chadda symbol looks like a lowercase “w” and is placed above a consonant to accentuate it.



(a) (b)

Figure 1.2: Example ligatures used in Arabic handwriting: a) regular ligature (upper with ligature, lower without); b) chadda ligature (upper with chadda, lower without)



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

Figure 1.3: Examples diacritics used in Arabic handwriting: a) Alif without diacritic; b) Alif with Hamza; c) Alif with Madda above; d) Alif with Madda below

This usually is an indicator of diacritics, optional symbols associated to one character. But Chadda is defined to form ligatures with its corresponding character (e.g., see Figure 1.2(b)).

Another difference between Arabic and English handwriting is the use of diacritics. Most diacritics do not matter for the written word and are thus optional, but they are used for the articulation of the word. Two different diacritics are commonly used (and annotated within the database that is used for the Arabic recognition task):

Hamza A “c”-like symbol which is used to mark an unvoiced glottal sound. Hamza can be placed above or below the character (e.g., see Figure 1.3(c) and Figure 1.3(d))

Madda A tilde, which can only be associated to the character “Alif” (except when used in the Koran), which marks the spoken concatenation with a second, long-spoken vocal (e.g., see Figure 1.3(b)).

An overview of the basic Arabic characters without the optional diacritics or ligatures is given in Table 1.1.

Table 1.1: Overview of Arabic characters (Char) and their annotation (Anno) in the IFN/ENIT database (see Chapter 4

Name	**Isolated (A)**		**Initial (B)**		**Middle (M)**		**Final (E)**	
	Char	**Anno**	**Char**	**Anno**	**Char**	**Anno**	**Char**	**Anno**
Alif/Alef	ا	aaA					ـا	aaE
Baa	ب	baA	بـ	baB	ـبـ	baM	ـب	baE
Taa	ت	taA	تـ	taB	ـتـ	taM	ـت	taE
Thaa	ث	thA	ثـ	thB	ـثـ	thM	ـث	thE
Jiim/Jeem	ج	jaA	جـ	jaB	ـجـ	jaM	ـج	jaE
Haa	ح	haA	حـ	haB	ـحـ	haM	ـح	haE
Khaa	خ	khA	خـ	khB	ـخـ	khM	ـخ	khE
Daal	د	daA					ـد	dae
Dhaal/Thaal	ذ	dhA					ـذ	dhE
Raa	ر	raA					ـر	raE
Zaay	ز	zaA					ـز	zaE
Siin/Seen	س	seA	سـ	seB	ـسـ	seM	ـس	seE
Shiin/Sheen	ش	shA	شـ	shB	ـشـ	shM	ـش	shE
Saad	ص	saA	صـ	saB	ـصـ	saM	ـص	saE
Daad/Shaad	ض	deA	ضـ	deB	ـضـ	deM	ـض	deE
Taa/Tta	ط	toA	طـ	toB	ـطـ	toM	ـط	toE
Dhaa/Dthaa	ظ	zaA	ظـ	zaB	ـظـ	zaM	ـظ	zaE
Ayn/Ain	ع	ayA	عـ	ayB	ـعـ	ayM	ـع	ayE
Ayn/Ain	غ	ghA	غـ	ghB	ـغـ	ghM	ـغ	ghE
Faa	ف	faA	فـ	faB	ـفـ	faM	ـف	faE
Qaaf/Qaf	ق	kaA	قـ	kaB	ـقـ	kaM	ـق	kaE
Kaaf/Kaf	ك	keA	كـ	keB	ـكـ	keM	ـك	keE
Laam/Lam	ل	laA	لـ	laB	ـلـ	laM	ـل	laE
Miim/Mem	م	maA	مـ	maB	ـمـ	maM	ـم	maE
Nuun/Noon	ن	naA	نـ	naB	ـنـ	naM	ـن	naE
Haa	ه	heA	هـ	heB	ـهـ	heM	ـه	heE
Waaw/Wow	و	waA					ـو	waE
Yaa/Yeh	ى	eeA					ـى	eeE

Chapter 2

State of the Art in Handwriting Recognition

Current research in HWR mainly follows two different basic approaches which will be discussed in this chapter. The first approach recognizes isolated characters or digits. Therefore, when dealing with non-isolated characters, for example in words or text, the data has to be segmented beforehand. The origin of this method has to be seen in the beginning of HWR when the input images contained only one single symbol.

The second approach uses a continuous recognition, meaning that beginning or ending of each character can be hypothesized at every position. This approach is widely used by HMM based recognizers, one of which is described in detail in Chapter 3.

The remainder of this chapter is organized as follows: Section 2.1 gives a brief overview of techniques used for isolated character or digit recognition, while Section 2.2 describes methods on preprocessing, feature extraction, modeling and decoding currently used in HMM based HWR systems.

2.1 Isolated Character or Digit Recognition

The task to be performed when dealing with isolated character or digit recognition is the classification of one input date as a character or digit. The recognition is performed, using one or several classifiers. A naive approach to the recognition of an image containing a single symbol is the nearest neighbour (NN) or k-NN approach (see [Keysers & Dahmen$^+$ 00, Keysers & Deselaers$^+$ 07]). More sophisticated algorithms which achieve better results than the naive NN are artificial neural networks (NNs) and support vector machines (SVMs) (for details on both see [Duda & Hart$^+$ 01]) as introduced by [Lee 96], and [Srihari 93]. A more comprehensive comparison of current techniques is, for example, given by [Keysers 07], and [Aburas & Gumah 08].

The recognition of isolated characters can be extended to text recognition by first segmenting the input image into characters or pieces of a character. Thereafter, an isolated character recognition is performed on these segments. The difficulty of this

approach is the segmentation step, as it is essential for the recognition results (see [Koerich & Sabourin$^+$ 03]).

The recognition of isolated characters or digits is not part of this thesis. Hence, this brief overview of such recognition methods shall suffice.

2.2 Continuous Word Recognition

Contrary to the isolated recognition, the continuous recognition does not require a segmented input image. Most of the common recognition systems are HMM based allowing for an hypothesization of the character and word boundaries at every position in the input image (see [Pechwitz & Märgner 03, Alma'adeed & Higgens$^+$ 02, Lorigo & Govindaraju 06, Märgner & Abed 07]). The most likely hypothesis is chosen in an optimization step together with the recognized characters. This process is described in more detail in Chapter 3. The task of continuous word recognition is to recognize an unknown number of words within one input sample.

According to [Bertolami & Bunke 08], current systems are able to achieve recognition rates between 50% and 80% depending on the recognition setup. Possible setups are, for example, line or sentence recognition. The former setup deals with the recognition of one text line as within a handwritten page. Thereby the difficulty is the recognition of hyphenated words at a line ending or beginning which increases the number of possible words. Moreover, the number of sentences within one line is irregular. Since sentences may not begin with the first and end with the last word in a line, most text corpora cannot be used to create LMs without further processing. The sentence recognition setup is a special case of line recognition, where one line contains exactly one sentence. The sentence recognition is easier than general line recognition since no hyphenation is expected and the punctuation marks always appear at the last position within a line. A third possible setup is single word recognition which can also be seen as a special case of line recognition, with each line containing one word. With this setup, recognition rates above 90% have been achieved (see [Märgner & Abed 07]).

When taking a look at current state of the art systems, the most important for the context of this work is the system presented by [Natarajan & Saleem$^+$ 08] at BBN Technologies. It is one of the few systems explicitly designed for script independent recognition. The presented system has been tested on a variety of databases, amongst others the IAM and the IFN/ENIT databases which are also used for this work and are described in more detail in Chapter 4. The respective system is able to achieve good results in both databases, the best word error rate (WER) on the IAM database is 40.1% and 10.6% on the IFN/ENIT database. Though the results are not obtained on the original evaluation folds of the databases these error rates can be seen as a generic benchmark.

Other systems focusing on western handwriting can achieve similar results. For example, the system described by [Juan & Toselli$^+$ 01] can achieve about 40% WER on the IAM database. The system focuses on integrated recognition and interpretation of the handwritten text. The system best performing on the IAM database by [Bunke & Bengio$^+$ 04] achieves about 35% WER with a recognition setup focusing on geometric feature extraction.

Another state of the art system being only evaluated on the IFN/ENIT database, was constructed by [Caesar & Gloger$^+$ 93, Schambach & Rottland$^+$ 08] . It is the best performing system of the ICDAR 2007 competition on the IFN/ENIT database, gaining about 13% WER. This commercial system makes use of different HMM topologies for the modeling of variants in handwriting.

Additional possibilities to improve these recognition results have been introduced in several papers. For example, [Bertolami & Bunke 08] use system combination to improve the WER achieved on the IAM database up to 3% absolute. A multi pass recognition technique for writer adaptation to improve the results on the IFN/ENIT database is proposed by [Dreuw & Rybach$^+$ 09].

The systems outlined above are described in more detail in the following subsections following the order of the processing steps which are preprocessing, feature extraction, modeling and decoding. In addition, a short overview of multi pass and adaptation methods is given.

2.2.1 Preprocessing

Most data for offline handwriting recognition is extracted from the scans of pages of handwritten text. The quality of these scanned pages is often poor due to scanning artifacts, noise or low resolution. From these text pages, text lines or single words have to be extracted for recognition, non-text background like figures and charts need to be ignored. Since the data used in this thesis is already provided as text lines or single words (see chapter 4), the subject of line extraction from scanned images is not discussed here. Possible approaches and recent discussions of line extraction from scanned images can be found in [Juan & Toselli$^+$ 01], [Shafait & van Beusekom$^+$ 08] and [Gupta & Niranjan$^+$ 06].

Depending on the image quality, the extracted text lines or words have to be preprocessed for further usage, for example, using deslanting for the correction of cursive writing or writing line estimations for the correction of skewed text and height normalization. Other examples are binarization and components analysis for estimation of partial words. These techniques will be described in this subsection.

Deslanting

In order to remove cursive writing resulting from handwriting, a slant correction or deslanting can be applied. The slant correction usually uses a shear operation along the x-axis of a given image. Deslanting is needed if the character modeling cannot compensate for variety in writing style.

A method introduced by [Sun & Si 97] uses histograms of gradients over the current image. By identifying a peak in the histogram, an angle for the shear transformation is found. A similar method using the sobel edge operator is introduced by [Yanikoglu & Sandon 98]. A second approach introduced by [Sun & Si 97] uses parallelograms that are fitted to each connected component within an image. Each component is then corrected for the angle between both main axes of the parallelogram.

A method used by [Bertolami & Bunke 08, Vinciarelli & Luettin 01] projects all gray scale intensities to the baseline, given a projection angle. The baseline is separated into regular non-overlapping windows. The angle of projection is than optimized for each window by minimizing the variance of the projected data within the windows. Once determined, the optimal angle for each image is used for the shear operation.

[Caesar & Gloger$^+$ 93] and [Pechwitz & Märgner 03] use only the contour of the text to calculate a contour polygon and further calculate on the angles of the polygon segments. A shear histogram is extracted on the angles using a modified Hough transform. The histogram is weighted according to the length of the corresponding angles' segments. The mean of the weighted histogram is used as shear angle. The shear angle is removed by applying this algorithm iteratively.

Baseline and Top-Minuscule Line Estimation

As most state of the art systems use a linear HMM topology, they are able to compensate for variations in writing direction. The variations in writing height are not be compensated by a linear HMM itself. Therefore, a preprocessing step can be used to normalize the height and position of the writing and its ascenders and descenders. Ascenders and descenders are those parts of a character that are drawn above the top-minuscule line (e.g., uppercase letters, “b” or “d”) or below the baseline (e.g., “p” or “y”). The two lines frame most lowercase characters and thus are the most important of the so-called writing lines.

One approach is the estimation of the writer's baseline and top-minuscule line to normalize the position of both lines. The two lines can also be used for rotation of the text passage, vertical scale normalization and the detection of ascenders and descenders.

A method proposed by [Caesar & Gloger$^+$ 93] uses the local minima of the lower margin for each word to find the baseline. Every minimum is weighted regarding the

local context such that peeks are weighted lower than flat local minima. Based on these local minima, a linear regression line is calculated and used as baseline. The top-minuscule line is calculated in a similar manner taking into account that more ascenders than descenders exist.

Another method introduced by [Vinciarelli & Luettin 01] makes use of horizontal density histograms on desloped images. Desloping equals deslanting but uses a rotation instead of a shear transformation to orient the writing horizontally. By using a threshold, the core region of a text line is identified and the baseline and top-minuscule lines are set below respectively above this core region. This technique is used by [Bertolami & Bunke 08], and [Bunke & Bengio$^+$ 04].

An alternative approach to the estimation of baseline and top-minuscule line is the height reduction of ascenders and descenders as proposed in [Juan & Toselli$^+$ 01]. This height reduction of ascenders and descenders additionally compensates for different heights of ascenders and descenders due to writing style. Two different styles and their height reduction can be seen in Figure 3.3 where the writer of the left text draws a shorter ascender of the “p” than the writer of the right text.

Binarization

Several preprocessing and feature extraction methods, as for example, the binary connected components analysis (BCC), need the data to be in binary form. Therefore, a common step in preprocessing is the binarization of images used, for example, by [Aburas & Gumah 08], and [Bertolami & Bunke 08]. The binarization can be achieved by applying an intensity threshold to the image.

In addition to the binarization, a thinning or skeletonization step can be applied. Both reduce the diameter but not the length of each binary component. The latter reduces the diameter to a minimum of one pixel. For examples of these techniques see [Aburas & Gumah 08], [Lorigo & Govindaraju 06], or [Caesar & Gloger$^+$ 93].

[Caesar & Gloger$^+$ 93] and [Mozaffari & Faez$^+$ 07] apply a BCC as proposed by [Mandler & Oberländer 89] to binary images. This method reduces the amount of data, but not of information, by representing all connected components in hierarchical order. This analysis is especially useful for the extraction of connected sub-words as introduced by [Mozaffari & Faez$^+$ 07].

For the analysis, the outermost components are extracted first, for example, the black contour of all words. Within each existing component all subcomponents are extracted recursively. For example, the lowercase character “b” consists of two components, the outermost “b” contour and the white filling of the loop. A single binary connected component consists of a contour polygon, color, the coordinates of the surrounding rectangle in the image, and a reference to their enclosed subcomponents, so that the complete image can be restored from the components.

Since the aim of this work is the improvement of modeling, only few preprocessing steps of the former mentioned will be applied. Deslanting as well as the size normalization will be used to compensate for variations in writing style as proposed by [Juan & Toselli$^+$ 01].

2.2.2 Feature Extraction

Based on the preprocessed input images, the feature extraction is performed. In most current systems this is done by shifting a window in writing direction over the image and extracting so-called frames at each window position with the window's content. The shift step is most commonly chosen so that these frames have an overlap. The window width varies depending on the scaling of the image and features that are to be extracted. All local features are extracted within the frames. Some systems divide the sliding window itself into several sub-windows and extract different features within each of the sub-windows (e.g., [Schambach & Rottland$^+$ 08, Bazzi & Schwartz$^+$ 99, Caesar & Gloger$^+$ 93, El-Hajj & Likforman-Sulem$^+$ 05, Juan & Toselli$^+$ 01]). The features themselves can be of various types and used in combination. The most commonly baseline features are listed below.

Appearance-based Features

The most basic type is the appearance-based feature which consist of the gray values of the input images' pixels either directly or after applying a filter method. Appearance-based features are used by [Dreuw & Jonas$^+$ 08], and [Juan & Toselli$^+$ 01].

Geometric Features

In contrast to appearance-based features, geometric features make no use of the pixel intensities or color values, but of their relationships to each other. A possible feature set, introduced by [Marti & Bunke 02b], makes use of nine geometrical features, extracted on binarized images with a one pixel wide window:

1. Number of foreground pixels
2. First order moment of the foreground pixels
3. Second order moment of the foreground pixels
4. Position of the upper contour
5. Position of the lower contour
6. First order derivative of the upper contour

7. First order derivative of the lower contour
8. Number of foreground-background transitions
9. Number of black pixels between the upper and lower contour

Structural Features

Another type of features that can be extracted are structural features. These features are not directly related to the geometry of the written text, but to the structure that forms a character. Possible features, as introduced by [Lorigo & Govindaraju 06], and [Caesar & Gloger$^+$ 93] are, for example, end-points of a stroke, dots or the strokes themselves.

The aim of this work is improved modeling for a script independent HWR system. Therefore, only appearance based features will be used since they are the most basic features and independent of the script used. Most of the other feature extraction methods are specialized and evaluated only on one script.

2.2.3 Modeling

Current HWR systems use a linear HMM for the modeling of characters, using time respectively writing direction as independent variable. Each state is associated with a Gaussian mixture density as state model (for more details on HMM system architecture see Chapter 3). The number of states differs depending on the resolution used for the feature extraction, and the topology of the HMM.

A non-linear HMM model is proposed by [Schambach & Rottland$^+$ 08], allowing for several writing variants within one model, for example, four variants for Latin handwriting: large or small, and block or cursive writing.

Model Length Estimation

Contrary to the most commonly used HMMs with a fixed number of states for all characters (e.g., by [Zimmermann & Bunke 02, Bazzi & Schwartz$^+$ 99]), a second approach uses MLE which models each character with a different number of states. MLE is implemented to gain a higher spatial resolution, as longer characters are modeled with more states than shorter characters.

[Schambach 03] describes a MLE method that iteratively changes the length of each character by +/-1 state per iteration by using an HMM with three linear submodels. One having the estimated number of states, one having a state more and

one a state less. At the end of an iteration, the sub model with the highest emission probability is chosen.

Two methods created for the use with HMM topology without skips are described by [Zimmermann & Bunke 02]. By using a topology without skips, the calculated number of states represents the minimum width of a character, shorter characters are not be modeled adequately. The first method extracts the mean length of all characters in a two pass step. The length of all character is then calculated by choosing a fraction and multiplying it with the mean length. The fraction is optimized regarding the WER. The second approach is similar to the first one but uses quantiles instead of calculating a fraction. [Dreuw & Jonas$^+$ 08] use the mean length of the characters directly for the length estimation but use an HMM topology with skips to allow the modeling of shorter characters.

Sub-words and Optional Characters in Arabic Handwriting

The recognition of Arabic handwriting needs special modeling due to the properties of the handwriting.

According to [Mozaffari & Faez$^+$ 07] and [Dreuw & Jonas$^+$ 08], the sub-word layout of Arabic handwriting (PAWs, see Section 1.2), and the difficult extraction of these sub-words, respectively the white spaces between them, requires a special treatment. [Mozaffari & Faez$^+$ 07] use a BCC to differ between PAWs, while [Dreuw & Jonas$^+$ 08] introduce an explicit whitespace modeling between PAWs and partial words (see Section 3.6.1 for examples).

Other specialties of Arabic writing are ligatures and diacritics (see Section 1.2). [Schambach & Rottland$^+$ 08] describes a special treatment for these optional symbols by adding special models for frequent ligatures or combinations of characters and diacritics.

Discriminative Training

Discriminative training can be used to model different writing styles writer independent. It is commonly used in speech recognition (see [Heigold & Deselaers$^+$ 08]) and has been applied with success to Thai HWR by [Nopsuwanchai & Povey 03]. An similar approach for discriminative training in Arabic HWR using the maximum mutual information (MMI) criterion was introduced by [Dreuw & Heigold$^+$ 09]. For further details on discriminative training see Section 3.4.3.

This works' focus is set on the improvement of modeling, therefore all of the mentioned methods are evaluated.

2.2.4 Decoding

Decoding in continuous recognition is generally divided into single and multi pass recognition. Single pass recognition uses one decoding step while multi pass recognition uses an iterative decoding to improve the results of each step based on information gained in a previous step.

Single Pass Recognition

Continuous text recognition often deals with the problem of unknown words in the test data. This is a realistic case since new words may occur at any time. Names, hyphenated, misspelled and crossed out words are most commonly unknown but likely in HWR. Additionally, a comprehensive lexicon increases the recognition complexity. Therefore, the recognition lexicon as well as the recognition LM have to be treated with special care. For example, so-called open lexica usually contain only a small number of most frequent words but are able to cover most of the testing data. A brief overview of the construction of open lexica is given in [Bunke & Bengio$^+$ 04].

Several methods for the generation and improvement of LMs for HWR are described in [Bunke & Bengio$^+$ 04], and [Pitrelly & Roy 03]. [Pitrelly & Roy 03] especially deal with hyphens within words and separations of a word at the end of a line.

Multi Pass Recognition

Multi pass recognition is usually used to improve the results iteratively, using information gained in a former recognition step to improve or modify a subsequent recognition step.

Hybrid Systems. Since handwriting has several syntactic levels (in descending order regarding their information content: words, hyphens, characters), recognition can be applied on any of these levels and can be refined on a higher level of information. A hybrid character- and word-level recognition is illustrated in [Bazzi & Schwartz$^+$ 99]. The system uses character recognition and then applies a word based LM for the recombination of the words from the recognized characters.

Writer Adaptation. Writer adaptation can be used to improve the recognition results by using writer dependent models (e.g., see [Chellapilla & Simard$^+$ 06] and [Dreuw & Rybach$^+$ 09]). These models can compensate for different writing styles, stroke thickness and other varieties resulting from different writers. The system proposed by [Dreuw & Rybach$^+$ 09] consists of a writer adaptive training process, while the decoding is done by two subsystems, each using differently trained models. In a

multi pass recognition, first a writer independent recognition is applied. In a second pass, the writer information are estimated by clustering the segments to be recognized. Writer adapted models are chosen using these writer information and then used for the second recognition pass.

System Combination. A method which is commonly used to improve results is system combination (see [Buck & Gass$^+$ 08]). This method makes use of the fact that different recognition systems make different errors. When using several recognition systems, the errors of each system can be compensated by the other systems. Various methods for the combination of HWR systems are given by [Bertolami & Bunke 05], [Bertolami & Bunke 08], and [Abed & Märgner 09].

This work uses only single pass recognition, thus, the main focus lies on the improvement of lexica and LMs.

Chapter 3

System Overview

In this chapter the system and methods used in this work are described.

This chapter is organized as follows: Section 3.1 gives a brief overview of the theoretical foundation of recognition and training setup. The subsequent sections describe the methods used for preprocessing (Section 3.2) and feature extraction (Section 3.3). The description of the applied modeling is divided into different sections on visual modeling (Section 3.4), language modeling (Section 3.5) and lexica (Section 3.6).

3.1 Theoretical Background

Current HWR systems use a statistical approach based on HMMs and Gaussian mixture models (GMMs). The input data is described by a sequence of features $x_1^T = x_1, ..., x_T$. By using the Bayes' decision rule introduced by [Bayes 63] which maximizes the a-posteriori probability, the best sequence of words $w_1^N = w_1, ..., w_N$ with unknown number of words N is optimized given $x_1^T = x_1, ..., x_T$:

$$[w_1^N]_{opt} = \arg\max_{w_1^N} \left\{ p(w_1^N | x_1^T) \right\} \tag{3.1}$$

$$= \arg\max_{w_1^N} \left\{ p^{\alpha}(w_1^N) \cdot p^{\beta}(x_1^T | w_1^N) \right\} \tag{3.2}$$

In Equation 3.2, the model is represented as $p(w_1^N) \cdot p(x_1^T | w_1^N)$. The term $p(w_1^N)$ is the LM which provides an a-priori probability for w_1^N. The term $p(x_1^T | w_1^N)$ is the visual model providing the probability of observing a sequence of features x_1^T given the word sequence w_1^N. Scaling factors for both models are applied with α being the LM scale and β being the visual model scale.

Figure 3.1 depicts the basic architecture of the used HWR system. The visual modeling, language models and the recognition are discussed in the following subsections.

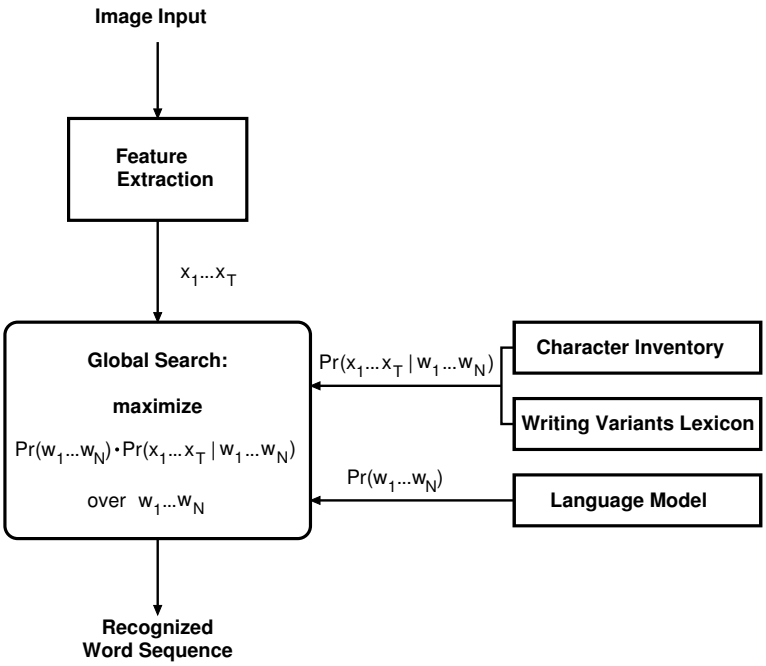


Figure 3.1: Recognition architecture of the proposed system

3.1.1 Visual Modeling

Visual models in HWR are used to provide stochastic models of text units. Usually, sub-word units such as characters are used and shared among all words in the vocabulary, allowing for the recognition of words not observed during the training. To create the model for a whole word the sub-word models are combined using a lexicon.

Hidden Markov Models

In the presented system the text units are modeled using linear HMMs which are able to compensate for variations in writing direction such as long drawn-out strokes. HMMs represent a text unit as a chain of ordered states. A probability distribution (mixture) is assigned to each state. The states perform as a stochastic finite state automata with different transitions from one state to another. In this work, the topology presented by [Bakis 76] is used. The Bakis topology has three transitions per state, a loop transition to stay in the current state, a forward transition to the next state in the chain and a skip transition to the next but one state. All regular character models use the Bakis topology except for the white space model that uses a similar topology without a skip transition and a single state only. An example of both topologies is given

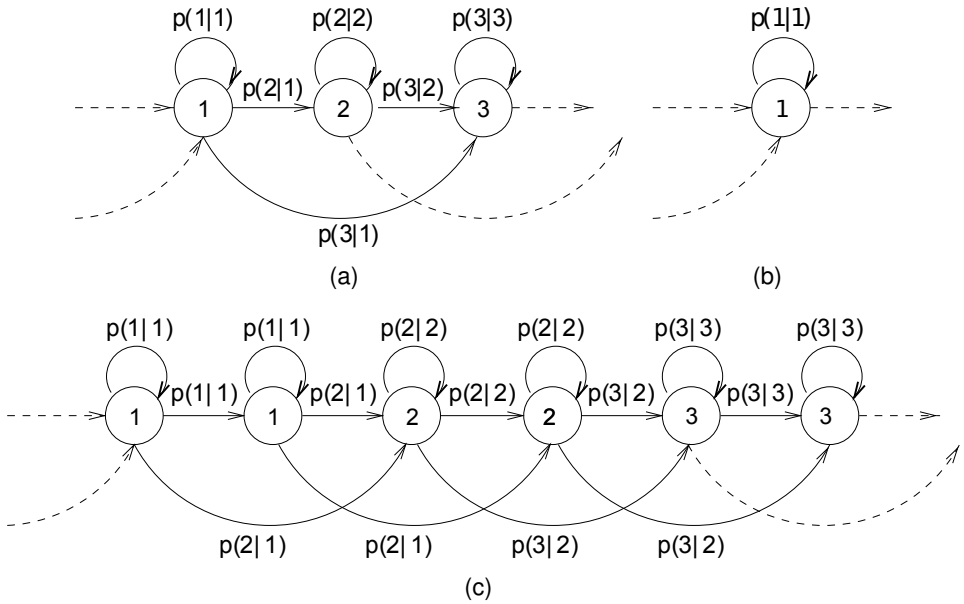


Figure 3.2: The HMM topologies used in this work: a) regular character models; b) white space model; c) character models with one repetition per state

in Figure 3.2. All states are associated with a unique mixture except when using state repetitions. Character models with state repetitions use the same transition topology but each state is repeated according to the number of repetitions. All repetition states share one mixture.

To estimate the probability $p(x_1^T|w)$ of observing the feature sequence x_1^T given a word w the sum over all possible state sequences s_1^T for this word has to be calculated.

$$p(x_1^T|w) = \sum_{[s_1^T]} p(x_1^T, s_1^T|w) \tag{3.3}$$

$$\tag{3.4}$$

with

$$p(x_1^T, s_1^T|w) = \prod_{t=1}^{T} p(x_t, s_t|x_1^{t-1}, s_1^{t-1}, w) \tag{3.5}$$

If sub-word units are used, the HMMs for the sub-word units are connected to a whole word model. The word models are then concatenated to model a word sequence. Using Equation 3.3 and Equation 3.5, the probability of observing x_1^T given w_1^N can

be calculated. It is assumed that the probability of observing x_t depends only on s_t which only depends on the preceding state s_{t-1} (first order Markov assumption). When applying a so-called Viterbi approximation as proposed by [Ney 90], the term can be rewritten as follows (see [Rybach 06] for the detailed equation):

$$p(x_1^T|w_1^N) = \sum_{s_1^T} \prod_{t=1}^{T} p(x_t, s_t|x_1^{t-1}, s_1^{t-1}, w_1^N) \tag{3.6}$$

$$\approx \max_{s_1^T} \left\{ \prod_{t=1}^{T} p(x_t|s_t, w_1^N) \cdot p(s_t|s_{t-1}, w_1^N) \right\} \tag{3.7}$$

In Equation 3.7, $p(x_t|s_t, w_1^N)$ is called emission probability. The term calculates the probability of observing the features x_t in state s_t, while $p(s_t|s_{t-1}, w_1^N)$ is the transition probability of moving from state s_{t-1} to state s_t.

In this work, the transition probability is assumed to be independent of the word model containing the transition, thus, the transition probability is replaced by fixed values depending on the transition length $s_t - s_{t-1}$. These values are called time distortion penaltys (TDPs).

Mixture Densities

The emission probability is modeled using GMMs. The mixture models consist of a weighted sum of Gaussian probability densities:

$$p(x|s, w_1^N) = \sum_{l=1}^{L_s} p(l|s, w_q^N) \cdot p(x|s, l, w_1^N) \tag{3.8}$$

$$= \sum_{l=1}^{L_s} c_{sl} \cdot \mathcal{N}(x|\mu_{sl}, \Sigma_{sl}, w_1^N) \tag{3.9}$$

with $\sum_{l=1}^{L_s} c_{sl} = 1$ and L_s is the number of densities in state s, c_{sl} the mixture weight and $\mathcal{N}(x|\mu, \Sigma)$ the normal distribution with mean μ and covariance Σ. For efficiency reasons, statistical independence of the features is assumed and thus, a diagonal covariance matrix is used. In addition, a pooled covariance matrix is used to avoid problems with covariance estimation, hence $\Sigma_{sl} = \Sigma$. Since the mixture probability is usually dominated by one density, a maximum approximation can be applied. The emission probability for a state of a single word w is:

$$p(x|s, w) = \max_l \{c_{sl} \cdot N(x|\mu_{sl}, \Sigma, w)\} \tag{3.10}$$

3.1.2 Training of the Visual Models

During the training of the mixtures, the parameters mean μ_{lsw}, mixture weight c_{lsw} and variance σ have to be estimated. To estimate the parameters, the maximum likelihood principle is applied using an expectation maximization (EM) algorithm. The training is performed using the following algorithm iteratively:

1. Estimate the best path, i.e. the best sequence of HMM states. This mapping of feature vectors to HMM states is called time alignment.

2. Collect the observations (feature vectors) for each state.

3. Estimate the model parameters for the emission probabilities

The algorithm terminates if the parameters are stable or a fixed number of iterations is reached. A linear segmentation is used as initial best path. In a second step the mixtures are optimized by splitting and merging the densities repeatedly. Further details are described in [Rybach 06].

3.1.3 Language Models

LMs are used to model text properties like syntax and semantic independently from the visual models. They provide an a-priori probability $p(w_1^N)$ of a word sequence w_1^N. Since the number of word sequences is unlimited, so-called m-grams are used according to the assumption that a word sequence follows a $(m-1)$-th order Markov process. This means a word w_n depends only on its $(m-1)$ predecessors $h_n := w_{n-m+1}^{n-1}$ called its history. The probability of a word sequence is calculated as follows:

$$p(w_1^N) = \prod_{n=1}^{N} p(w_n | w_1^{n-1}) \tag{3.11}$$

$$= \prod_{n=1}^{N} p(w_n | h_n) \tag{3.12}$$

To evaluate an LM the perplexity is commonly used. It can be interpreted as the average number of possible words at each position in the text regarding the context.

An LM is trained using word counts of the transcripted training data or text corpora. Since not all possible m-grams can necessarily been seen in training and can therefore not be recognized, a smoothing is used to allow for unseen m-grams as proposed by [Kneser & Ney 95].

3.1.4 Recognition

The problem of finding the best word sequence w_1^N for a feature sequence x_1^T using the models described in the previous sections can be solved using Viterbi-search. All possible word sequences and their paths of states are hypothesized at all time steps. The HMM states of all hypotheses are expanded by calculating the set of successors for each state. The so-called pruning discards unlikely hypotheses during the calculation.

3.2 Preprocessing

This section describes the preprocessing applied in this work. The same preprocessing as proposed by [Juan & Toselli$^+$ 01] is used and can be divided into three main steps:

1. **Color normalization:** In the first step, the colors of the input images are normalized for a better image quality. Since the input image is supposed to be in gray scale format, only a remapping of the intensity values has to be applied. First, a gray scale spread is applied, mapping most of the intensity values to white to compensate for noise and background color in the image. Next, the image is filtered using a median filter. The median filter is a non-linear hierarchy filter replacing a pixel with the median value of its neighborhood.

2. **Deslanting:** In a second step the images are deslanted to remove cursive writing. The image is segmented into text segments at separating white spaces. On each segment, the deslanting proposed by [Yanikoglu & Sandon 98] is applied. It calculates the slant angle by computing slant histograms using Sobel edge operators. Therefore, the image is convolved with the Sobel edge operators to receive the horizontally $h(x, y)$ and vertically $v(x, y)$ filtered images. Then, the magnitude mag and phase $phase$ of the image are calculated according to the following equations:

$$mag(x, y) = \sqrt{v(x, y)^2 + h(x, y)^2} \tag{3.13}$$

$$phase(x, y) = \arctan\left(\frac{v(x, y)}{h(x, y)}\right) \tag{3.14}$$

   The slant histogram is computed by summing the magnitude of each slant angle $phase$ across the segment. The histogram is finally smoothed with a Gaussian filter and the maximum angle of the smoothed histogram between -40 and +40 degree to the vertical is chosen as slant angle. The restriction of the slant angle is used to prevent unlikely deformations.

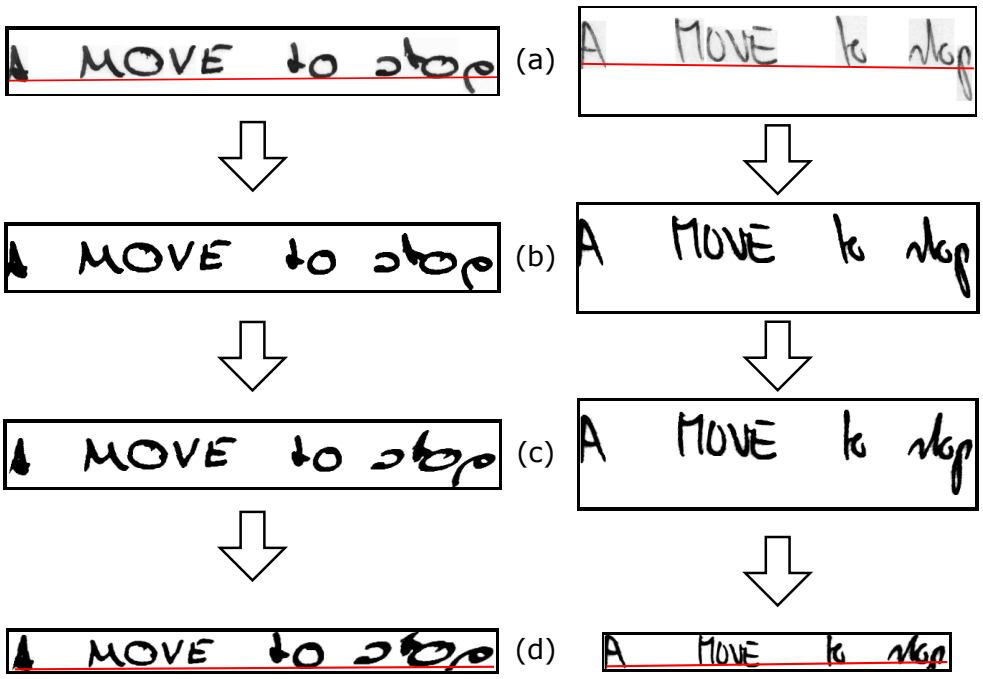


Figure 3.3: Preprocessing for two different example images, each in one column, from the IAM database: a) Original images; b) color normalized images; c) deslanted images; d) size normalized images. The red lines indicate the writing baseline.

3. **Size Normalization:** In a third step, ascenders and descenders are eliminated since they often vary in height or position and a linear HMM is not able to compensate for these variations. The main text body without ascenders or descenders is extracted from the image using reference lines extracted from the upper and lower contour. The image zones containing ascenders are scaled linear to a height of 30% of the text body respectively 15% for the zone containing descenders. The rescaled zones are then reattached to the main text body.

In Figure 3.3 the preprocessing on example images taken from the IFN/ENIT database is illustrated.

3.3 Feature Extraction

Since the main focus of this work is set on modeling, only appearance based features are used. The features are extracted using a sliding window which is shifted over the

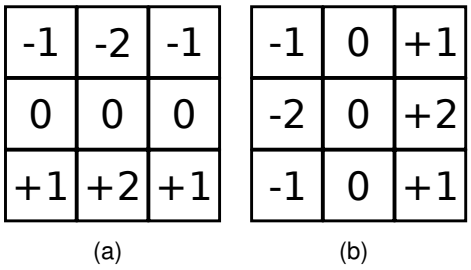


Figure 3.4: The horizontal (a) and vertical (b) Sobel filter

image in writing direction. Since the number of appearance based features depends on the height of the sliding window, all images are scaled to the same height. The shift width of the sliding window is one pixel, thus, the extracted frames are overlapping. The features are reduced using either principal components analysis (PCA) or linear discriminant analysis (LDA). Further explanations of the feature reduction methods are given in [Duda & Hart$^+$ 01].

Two feature sets are evaluated:

Motion
: The motion feature set contains the original intensity values within the sliding window and the spatial derivatives in horizontal direction $\Delta = x_t - x_{t-1}$.

Sobel
: The Sobel feature set contains the original and Sobel filtered intensity values. The Sobel filters are two filters that expose horizontal or vertical borders. The filters are depicted in Figure 3.4 Both the horizontal and vertical Sobel filtered values and the absolute values of the filtered image are used for the feature set. The absolute values are used to add values that are not linear independent of the original intensity values.

The feature extraction of both feature sets is illustrated in Figure 3.5.

Sobel filtered images have successfully been used for isolated handwritten digit recognition by [Keysers & Deselaers$^+$ 07]. Therefore, the transferability of Sobel filters to continuous HWR is evaluated in this work. The Motion feature set was introduced by [Dreuw & Jonas$^+$ 08] and has proven to achieve good results in single word recognition on Arabic handwriting.

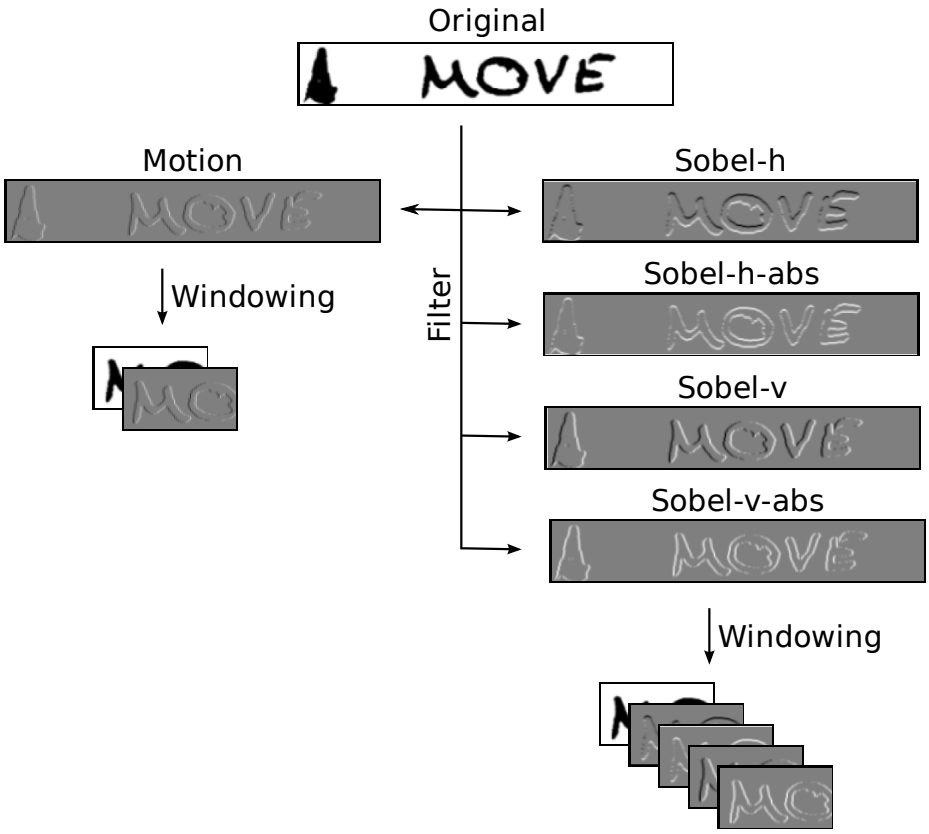


Figure 3.5: Example feature extraction for Motion feature set (left) and Sobel feature set (right) on an example image taken from the IAM database. Motion filter is the first derivative in writing direction; Sobel-h and Sobel-v the horizontal and vertical Sobel filters; Sobel-h-abs and Sobel-v-abs are the horizontal and vertical absolute values of the sobel filter.

3.4 Visual Modeling

The main focus of this work is the improvement of visual models. Therefore, several techniques known from HWR and speech recognition are evaluated. The MLE of visual models is described in Section 3.4.1. Section 3.4.2 describes an approach for context dependent parameter reduction known from speech recognition. For further improvement of the visual models discriminative training is applied as explained in Section 3.4.3.

3.4.1 MLE

MLE describes the adjustment of the model length, in order to improve the models' spatial resolution. For example, the handwritten character "M" is commonly much longer than the character "i". The visual models for these characters should represent these differences in length to obtain a similar spatial resolution for both characters. When using MLE, the number of HMM states used for each character is chosen with respect to the character length in a way that long characters have more HMM states.

Since the lengths of the characters are usually unknown, they have to be estimated. In this work, the method proposed by [Dreuw & Jonas$^+$ 08] is used which uses a two-pass training. In the first training pass, an alignment on the training data is estimated. This alignment is used to calculate the average length of each character seen in training. The estimated number of states S_c for a character c is calculated using the following equation:

$$S_c = \frac{N_{x,c}}{N_c} \cdot f_P \tag{3.15}$$

where $N_{x,c}$ is the number of observations aligned to character c, N_c is the count of character c seen in training and f_P is a scaling factor. Since the RWTH-ASR software[1] used for this work only allows for models with a fixed number of states, additional states are applied by substituting each character by pseudo-characters whose states sum up to the desired number.

3.4.2 CART

Experiments in speech recognition showed that the reduction of model parameters, so-called state tying, using context dependent models can improve the results obtained (see [Beulen & Bransch$^+$ 96]). Yet, current HWR systems do not use context dependent models or other context depending parameter reduction techniques. Therefore, this work evaluates the usage of CART for context based model parameter reduction in handwriting recognition as proposed by [Beulen & Bransch$^+$ 97] for

[1] http://www-i6.informatik.rwth-aachen.de/rwth-asr

speech recognition. A character with its left and right context of one character each will be called triphone as used in speech recognition. A triphone is annotated as ${}_{a}b_{c}$, for character b with the left context of a and the right context of c.

State tying is used to reduce the number of free parameters in order to estimate the remaining parameters more robustly. Here CART, a binary decision tree, without ad-hoc subdivision is used. The internal nodes of the tree are tagged with questions on the triphone context and type of the central character of the triphone. Possible questions are, for example, “Is the central character uppercase?”, or “Is the left context a word boundary?”. The leaves of the tree are tagged with labels identifying a mixture model of the context depending model.

To find the mixture for a triphone state, starting from the root node, recursively the questions tagged at the current node are asked and if the answer is “yes”, the left child of the current node is chosen, otherwise the right. When reaching a leaf, the appropriate mixture is identified by the mixture label.

The tree is constructed starting with the root node modeling all states of all possible triphones. The observations of a node are modeled by one Gaussian mixture density. Each leaf F of the tree is then splitted consecutively with the question giving the largest local improvement in likelihood $D(L, R)$ with the child nodes L and R:

$$D(L,R) = LL(F) - (LL(L) + LL(R)) \tag{3.16}$$

$$= -\frac{1}{2}\left(n_L \sum_{d=1}^{D} log\left[\frac{\sigma_{d,F}}{\sigma_{d,L}}\right]^2 + n_R \sum_{d=1}^{D} log\left[\frac{\sigma_{d,F}}{\sigma_{d,R}}\right]^2 \right) \tag{3.17}$$

with n_X being the number of observations for node X, D the dimensionality of the feature vector and $\sigma_{d,X}$ the variance of component d for node X. The algorithm terminates if a certain number of leaves is reached or if the likelihood of all leaves is below a certain threshold. For further details on the calculation see [Beulen & Bransch$^+$ 96].

3.4.3 Discriminative Training

To handle different writing styles and their variation in HWR the discriminative training based on a modified MMI introduced by [Heigold & Deselaers$^+$ 08] and and evaluated on Arabic HWR by [Dreuw & Heigold$^+$ 09] is used.

When using maximum likelihood training, the likelihood of the aligned training data $x_1^{T_r}$ given a transcription w_1^N for all aligned training data $1..R$ is maximized:

$$\mathcal{F}(\theta) \quad = \sum_{r=1}^{R} \log p_\theta(x_1^{T_r} | w_1^{N_r}) \tag{3.18}$$

The MMI criterion is most commonly referred to as the maximum likelihood for the

class posteriors:

$$\mathcal{F}^{(\text{MMI})}(\theta) \quad = -\frac{1}{N}\sum_{r=1}^{R}\log\left(\frac{p_\theta(x_1^{T_r}|w_1^{N_r})p(w_1^{N_r})}{\sum_{v_1^{M_r}} P_\theta(x_1^{T_r}|v_1^{M_r})p(v_1^{M_r})}\right) \tag{3.19}$$

with v_1^M being a writing variants lexicon. [Dreuw & Heigold$^+$ 09] define a modified MMI criterion (M-MMI):

$$\mathcal{F}_\gamma^{(\text{M-MMI})}(\theta) = \mathcal{R}(\theta,\theta_0) - \frac{J}{R}\sum_{r=1}^{R}\frac{1}{\gamma}\log\left(\frac{[p_\theta(x_1^{T_r}|w_1^{N_r})p(w_1^{N_r})\mathrm{e}^{(-\rho\delta(w_1^{N_r},w_1^{N_r}))}]^\gamma}{\sum_{v_1^{M_r}}[p_\theta(x_1^{T_r}|v_1^{M_r})p(v_1^{M_r})\mathrm{e}^{(-\rho\delta(w_1^{N_r},v_1^{M_r}))}]^\gamma}\right) \tag{3.20}$$

with $\mathcal{R}$ being a regularization constant proportional to $\frac{1}{J}$ and γ being a parameter to control the smoothness of the criterion called approximation level. $\mathrm{e}^{(-\rho\delta(w_1^{N_r},v_1^{M_r}))}$ is an additional margin term which is non-zero only for correct w_1^N. According to [Heigold & Deselaers$^+$ 08] $\mathcal{F}_\theta^{(\text{M-MMI})}(\Lambda)$ is a smooth approximation to an SVM with hinge loss function and can be optimized iteratively using gradient-based optimization techniques. In this work RProp is used for the optimization (see [Zhang & Jin$^+$ 03]).

3.5 Language Model

When creating LMs for handwritten line recognition tasks some special cases need to be taken into account. In general all text corpora composed for LM generation are compiled for sentence recognition as shown in Figure 3.6. This means that all text lines contain a single sentence, starting with the sentence beginning and ending with a potential punctuation mark and the sentence ending. A generated LM will therefore have high probabilities for punctuation marks at the line ending but low probabilities for punctuation marks within the line. But when recognizing a handwritten text line sentences may begin or end at any position and the number of partial sentences per line may vary. Thus, the text corpora prepared for sentence recognition need to be preprocessed. In this work a simple approach to simulate line writing is applied by concatenating all sentences and inserting new lines at a fixed interval of words. Several intervals have been tested and a number of 10 words per line were chosen empirically which increases the number of lines in the corpus by a factor of about two. Figure 3.7 shows lines from a preprocessed corpus while Figure 3.8 shows the same passage with explicit white spaces.

Another specialty of handwritten line recognition is hyphenation which divides the last word of a line into two new words. Hyphenation may result in two new words to be recognized which are likely not to be observed in the training data, since parts of hyphenated words are not necessarily a valid word themselves. Nonetheless, this

```
<line>Delegates␣from␣Mr.␣Kenneth␣Kaunda's␣United␣National
␣independence␣party␣(␣280,000␣members␣)␣and␣Mr.␣Harry
␣Nkumbula's␣African␣National␣congress␣(␣400,000␣)␣will␣meet
␣in␣London␣␣today␣to␣discuss␣a␣Common␣course␣of␣action␣.
</line>
<line>Sir␣Roy␣Is␣violently␣opposed␣to␣Africans␣getting␣an
␣elected␣majority␣␣in␣Northern␣Rhodesia␣,␣but␣the␣colonial
␣Secretary␣,␣Mr.␣Iain␣Macleod␣,␣is␣insisting␣on␣a␣policy␣of
␣change␣.</line>
```

Figure 3.6: Example text lines from the LOB corpus without further preprocessing. "␣" marks possible white spaces.

work will not apply a special approach for hyphenated words because only very few cases of hyphenations appear in the databases.

All LMs are created using the SRILM toolkit (see [Stolcke 02]).

3.6 Lexica

The creation of word lexica/vocabulary for continuous HWR requires a special treatment. The lexica contain the words that can be recognized and their annotation corresponding to the visual model. In this work, only character models are used, therefore a word lexicon contains a character annotation. Since the number of words that can be recognized is depending on the lexicon, size and contained words need to be chosen carefully. Words not in the lexicon, so-called out of vocabulary (OOV) words, cannot be recognized. Thus, a small lexicon may lead to bad recognition results, since less words can be recognized but a large lexicon increases the computation time and more search errors can occur. Therefore, recognition lexica for continuous text recognition are created using the most frequent words from large text corpora.

Additional improvements of the lexica are described in more detail in the following subsections.

3.6.1 White Space Modeling

As proposed by [Dreuw & Jonas$^+$ 08], an explicit white space (blank) modeling is used for Arabic handwriting instead of modeling white spaces implicitly in the character models. The explicit whitespace models improve the model quality of the character models since less white space needs to be modeled implicitly. In addition, compound

```
<line>...␣Delegates␣from␣Mr.␣Kenneth␣␣Kaunda's</line>
<line>United␣National␣independence␣party␣(␣280,000
␣members␣)␣and␣Mr.</line>
<line>Harry␣Nkumbula's␣African␣National␣congress␣(␣400,000␣)
␣will␣meet␣in␣London␣␣today␣to␣discuss␣a␣Common␣course␣of
␣action␣.␣Sir</line>
<line>Roy␣Is␣violently␣opposed␣to␣Africans␣getting␣an
␣elected␣majority</line>
<line>in␣Northern␣Rhodesia␣,␣but␣the␣colonial
␣Secretary␣,</line>
<line>Mr.␣Iain␣Macleod␣,␣is␣insisting␣on␣a␣policy␣of</line>
<line>change␣.␣...</line>
```

Figure 3.7: Example text lines from the LOB corpus with automatic line breaks every 10 words. Optional white spaces are marked with "␣".

words respectively sub-words that are divided by explicit white spaces are better separated since the borders to white spaces are easier to detect than the border between two characters. Possible white space models are:

No white spaces (NB)
: Using the implicit whitespace modeled by the character models.

Between word white spaces (BWB)
: White spaces are modeled explicitly between compound words. The NB model is added as writing variant.

Between word and within word white spaces (BWWB)
: White spaces are modeled explicitly between compound words and between PAWs respectively between initial, final and isolated characters. The BWB and NB models are added as writing variants.

Figure 3.9 shows an example of the possible white space positions.

3.7 Sub-word Lexica and Language Models

The rate of OOV words in the testing data can be seen as the lowest possible error rate. To further improve the results without increasing the data used for the lexicon generation, a recognition on a sub-word level can be performed. The usage of sub-words decreases the OOV rate, since sub-words are shared between several words.

```
<line>... [blank] Delegates [blank] from [blank] Mr. [blank]
 Kenneth  [blank] Kaunda's</line>
<line>United [blank] National [blank] independence [blank]
 party [blank] ( [blank] 280,000 [blank] members [blank] )
 [blank] and [blank] Mr.</line>
<line>Harry [blank] Nkumbula's [blank] African [blank]
 National [blank] congress [blank]( [blank]400,000 [blank] )
 [blank] will [blank] meet [blank] in [blank] London  [blank]
 today [blank] to [blank] discuss [blank] a [blank] Common
 [blank] course [blank] of [blank] action [blank] . [blank]
 Sir</line>
<line>Roy [blank] Is [blank] violently [blank] opposed
 [blank] to [blank] Africans [blank] getting [blank] an
 [blank] elected [blank] majority</line>
<line>in [blank] Northern [blank] Rhodesia [blank] , [blank]
 but [blank] the [blank] colonial [blank] Secretary [blank]
 ,</line>
<line>Mr. [blank] Iain [blank] Macleod [blank] , [blank] is
 [blank] insisting [blank] on [blank] a [blank] policy
 [blank] of</line>
<line>change [blank] . [blank] ...</line>
```

Figure 3.8: Example text lines from the LOB corpus with automatic line breaks every 10 words. Required white spaces are marked with "[blank]".

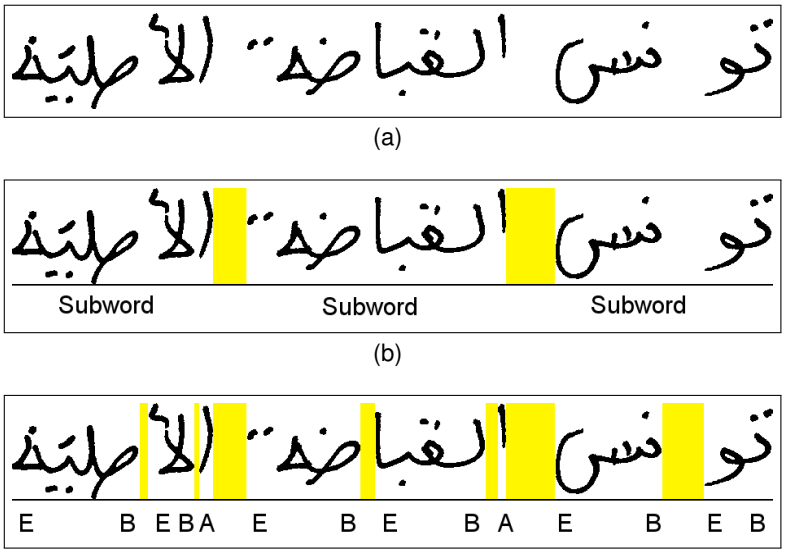


Figure 3.9: Example image with explicit white space (yellow) modeling: a) no explicit white spaces (NB); b) white spaces between sub-words (BWB); c) white spaces between PAWs respectively between initial (B), final (E) and isolated (A) characters (BWWB).

Thereby the number of necessary items in the vocabulary is decreased while the number of possible words is increased. For example, 20,000 words consist of only about 13,000 unique syllables or of 80 unique characters with respect to lower- and uppercase characters, and other symbols. Therefore, lower error rates are possible. The complete words can be recombined in a postprocessing step. Though when using sub-word lexica and LMs, the context information of a sub-word regarding its predecessors is smaller compared the context of a complete word. Thus, a higher number of predecessors have to be taken into account when calculating n-gram LMs.

In this work, sub-word units of syllable and character level are tested. In order to recombine the words from the sub-words, an explicit white space model is used as indicator for word boundaries.

To create a sub-word lexicon and LM the original words are splitted into sub-words. The character lexica are created as obvious by dividing the word into its characters (e.g., see Figure 3.10). For the creation of the syllable lexica a hyphenation software[2] is used. The software used calculates the possible position of hyphenations using the same technique as LaTeX as described by [Liang 83]. Unfortunately, the software is not able to calculate all possible positions but neither was any other of the tested methods.

[2] Text-Hyphen, `http://search.cpan.org/~kappa/Text-Hyphen-0.11/`

```
<line>... [blank] D e l e g a t e s [blank] f r o m [blank]
 M r . [blank] K e n n e t h [blank] K a u n d a ' s</line>
<line>U n i t e d  [blank] N a t i o n a l [blank] i n d e
 p e n d e n c e [blank] p a r t y [blank] ( [blank] 2 8 0
 , 0 0 0 [blank] m e m b  e r s [blank] ) [blank] a n d
 [blank] M r .</line>
<line>H a r r y [blank] N k u m b u l a ' s [blank] A f r
 i c a n [blank] N a t i o n a l [blank] c o n g r e s s
 [blank]( [blank] 4 0 0 , 0 0 0 [blank] ) [blank] w i l l
 [blank] m e e t [blank] i n [blank] L o n d o n  [blank]
 t o d a y [blank] t o [blank] d i s c u s s [blank] a
 [blank] C o m m o n [blank] c o u r s e [blank] o f
 [blank] a c t i o n [blank] . [blank]  S i r</line>
<line>R o y [blank] I s [blank] v i o l e n t l y [blank]
 o p p o s e d [blank] t o [blank] A f r i c a n s [blank]
 g e t t i n g [blank] a n [blank] e l e c t e d [blank]
 m a j o r i t y</line>
<line>i n [blank] N o r t h e r n [blank] R h o d e s i a
 [blank] , [blank] b u t [blank] t h e [blank] c o l o n
 i a l [blank] S e c r e t a r y [blank] ,</line>
<line>M r . [blank] I a i n [blank] M a c l e o d [blank]
 , [blank] i s [blank] i n s i s t i n g [blank] o n
 [blank] a [blank] p o l i c y [blank] o f</line>
<line>c h a  n g e [blank] . [blank] ...</line>
```

Figure 3.10: Example text lines for a character lexicon and LM from the LOB corpus with automatic line breaks every 10 words. Required white spaces are marked with "[blank]".

```
<line>... [blank] Del e gates [blank] from [blank] Mr.
 [blank] Ken neth  [blank] Kaun da's</line>
<line>Unit ed [blank] Na tion al [blank] in de pen dence
 [blank] par ty [blank] ( [blank] 280,000 [blank] mem bers
 [blank] ) [blank] and [blank] Mr.</line>
<line>Har ry [blank] Nkum bu la's [blank] African [blank]
 Na tion al [blank] congress [blank]( [blank]400,000 [blank]
 ) [blank] will [blank] meet [blank] in [blank] Lon don
 [blank] to day [blank] to [blank] dis cuss [blank] a [blank]
 Com mon [blank] course [blank] of [blank] ac tion [blank] .
 [blank] Sir</line>
<line>Roy [blank] Is [blank] vi o lent ly [blank] op posed
 [blank] to [blank] Africans [blank] get ting [blank] an
 [blank] elect ed [blank] ma jor i ty</line>
<line>in [blank] North ern [blank] Rhode sia [blank] ,
 [blank] but [blank] the [blank] colo nial [blank] Sec re
 tary [blank] ,</line>
<line>Mr. [blank] Iain [blank] Macleod [blank] , [blank] is
 [blank] in sis ting [blank] on [blank] a [blank] pol i cy
 [blank] of</line>
<line>change [blank] . [blank] ...</line>
```

Figure 3.11: Example text lines for a syllable lexicon and LM taken from the LOB corpus with automatic line breaks every 10 words. Required white spaces are marked with "[blank]".

Chapter 4

Corpora

For evaluation of this thesis' proposed methods two databases and several text corpora which are widely used in current literature and were chosen.

The IFN/ENIT database provides a set of Arabic handwritten words for single word recognition. For evaluation of continuous handwriting, the IAM database contains lines of handwritten English text. Results obtained on these databases are given in Chapter 5. Some of the LMs and lexica used for recognition are created using the additional text corpora.

4.1 IFN/ENIT Database

The IFN/ENIT database was published by the "Institut of Communications Technology" (Institut für Nachrichtentechnik, IFN) at the Technical University Braunschweig, Germany, and the "Ecole Nationale d'Ingénieurs de Tunis" (ENIT), Tunisia in 2002 (see [Pechwitz & Maddouri$^+$ 02]).

The database contains a total number of 306 different annotated symbols. These symbols include the 28 basic Arabic characters, respectively their variants based on the position in the word, combined with diacritics and a number of ligatures. It contains about 32,000 binarized images, each depicting a Tunisian town name. The images are classified by the zip code of the associated town with a total of about 1,000 classes. In addition to the zip code, all images are annotated with their spelling,

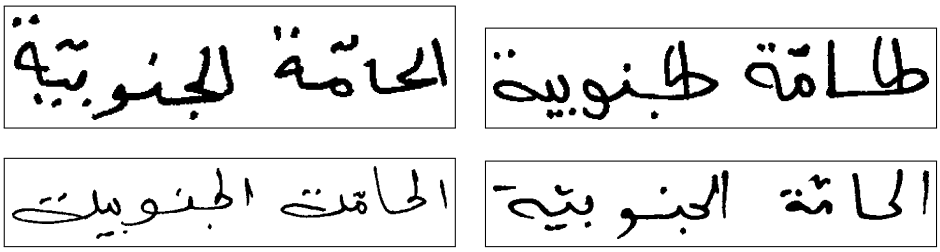


Figure 4.1: Sample images taken from the IFN/ENIT database (same word)

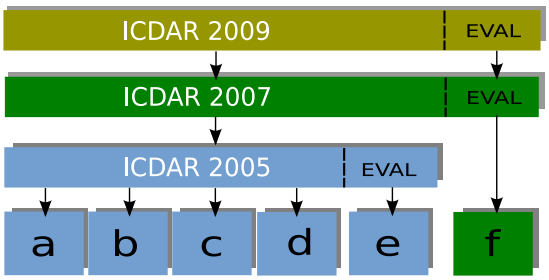


Figure 4.2: History of development of the IFN/ENIT database

Table 4.1: Description of the IFN/ENIT database

	Devel a	Devel b	Devel c	Devel d	Eval e	Total
Samples (Words)	6537	6710	6477	6735	6033	32492
Running Characters	51986	53863	52156	54167	45194	257366
Writers	102	102	103	104	505	916

including diacritics and ligatures. Due to writing variants of some town names, about 3,000 different notations exist (three different writing variants for each town name in average). As displayed in Table 4.1, the database is divided into five folds. Four of these folds are used as cross validation folds, whereas the remaining fold serves as evaluation fold. The reason for this division is given by the database's history of development as displayed in Figure 4.2. The database was first published as part of the ICDAR (International Conference on Document Analysis and Recognition) 2005 Arabic handwriting recognition competition (see [Märgner & Pechwitz$^+$ 05]) with four cross validation folds (Fold a-d) but without set e. For evaluation of the competition, participants had to submit their trained systems which were tested on the evaluation fold e. For the ICDAR 2007 (see [Märgner & Abed 07]) competition the e fold was published and the systems were later on evaluated on an unpublished evaluation set f. For the ICDAR 2009 competition no new dataset was published. Some sample images of the database are depicted in Figure 4.1. The database's task is a closed single word recognition, meaning that all words and characters in the test or evaluation sets appeared in the corresponding training data. Table 4.1 indicates that the training and testing folds are not equally distributed. Especially the number of different writers in fold e is higher than in all other folds which might cause a higher error rate compared to the sets with less writers.

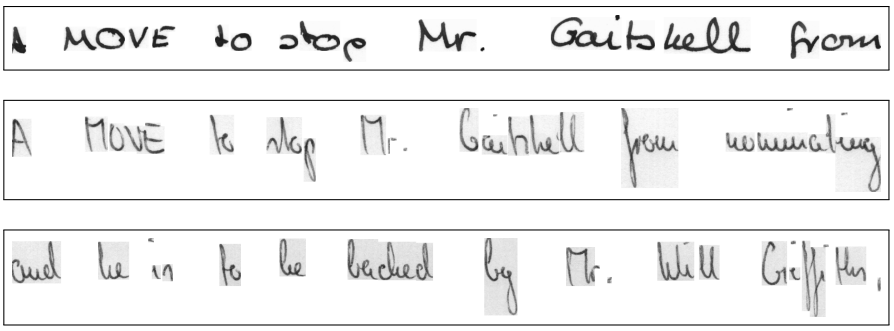


Figure 4.3: Sample images taken from the IAM database

4.2 IAM Database

The IAM database (see [Marti & Bunke 99]) was published by the Institute of Computer Science and Applied Mathematics (Institut für Informatik und angewandte Mathematik, IAM) at the University of Bern. The current database version 3.0 was introduced by [Marti & Bunke 02a] in 2002. It contains a total number of 1,539 pages with 5,685 sentences in 9,862 lines. All words are build using only 80 different symbols which consist of both upper- and lowercase characters, punctuation, quotation marks, and a special symbol for crossed out words. A comparison of the predefined training, testing and evaluation folds is given in Table 4.2, whereas Figure 4.3 displays three lines of same text written by different writers. According to [Marti & Bunke 02a] and as the figure implies, the images published with the database have been preprocessed. All connected components were extracted with their bounding boxes and then pasted on a virtual writing line. The image was then automatically cropped to minimum width and height. Thus, the image height and the relative height of the writings baseline and top-minuscule line differs from image to image, depending on the writing height of ascenders or descenders, respectively their absence. According to the paper, the applied preprocessing should not have an effect on the recognition results.

Four different tasks are provided with the database: a writer identification task and three text recognition tasks, each a word, sentence and line recognition task. In this thesis the focus is on the line recognition task, as it is the most basic approach to continuous HWR, and differs from other tasks for example by segment boundaries within sentences and syllabification.

As underlying corpus of the database, the Lancaster-Oslo/Bergen (LOB) corpus, is described in detail in Section 4.3. Thus all sentences occurring in the IAM database were taken from the LOB corpus.

Table 4.2: Description of the IAM database

	Train	Eval	Devel	Total
Lines	6,161	2,781	920	9,862
Running words	53,884	2,5473	8,718	88,073
Vocabulary size	7,764	5,312	2,425	11,368
Running Characters	281,744	126,239	42,074	450,047
Unique Writers	283	162	57	500
OOV Rate		15.45%	15.03%	

Table 4.3: Description of the text corpora

	Brown	LOB	Wellington
Lines	49,362	53,694	56,745
Running words	1,045,213	1,119,375	1,144,401
Vocabulary size	53,115	48,276	58,919
Running Characters	5,582,023	5,909,273	6,055,820

4.3 Text Corpora

When performing an open word recognition task, the training data provided by the database is most often not sufficient to model the data to be recognized. Therefore, additional text corpora are often needed to create LMs and lexica for the recognition task. The corpora used in this thesis follow the proposition by [Bertolami & Bunke 08] and can be described as follows.

The Brown corpus was published by [Francis & Kucera 64] in 1964. It was the first modern computer readable corpus and contains about one million words in 500 lines of American English from various text sources, such as reportage, humor and fiction.

The LOB corpus (see [Johansson & Leech$^+$ 78]) was compiled by researchers from Lancaster, Oslo and Bergen between 1970 and 1978. It corresponds to the Brown corpus in size and content, but contains words in British English.

The third corpus, which also equals the LOB corpus in size and content is the Wellington corpus, published by [Holmes & Vine$^+$ 98], and contains words in New Zealand English.

Table 4.3 gives a more comprehensive overview of the three text corpora.

Chapter 5

Experiments and Results

In this chapter the main experimental results on both the IFN/ENIT and the IAM database are reported using the metrics explained in Section 5.1. Since the languages as well as the tasks performed on the databases differ, the results for the Arabic and English HWR will be displayed in unique sections. The results of the single word recognition task on the IFN/ENIT database are discussed in Section 5.2. Section 5.3 contains the evaluation of the line recognition task on the IAM database.

5.1 Metrics and Visualization

The error rates commonly used for the evaluation of continuous text recognition are the word error rate (WER) and character error rate (CER). Both are calculated based on the number of substitutions, insertions and deletions of words respectively characters and the number of words or characters in the reference text using the following equation:

$$ER = \frac{\#\text{substitutions} + \#\text{insertions} + \#\text{deletions}}{\#\text{referencecount}} \tag{5.1}$$

The alignment visualizations used in this chapter show training alignment of text lines to their corresponding HMM states. A R-G-B background colors scheme is used for the 0-1-2 HMM states in writing direction and yellow for the white space model. The upper line contains the character model names, where the white space models are annotated by 'si' for 'silence'. The state numbers are written in the bottom line. HMM state-loops are represented by no-color-changes, forwards from one color to the next and skips by the change to the next but one.

5.2 IFN/ENIT Database

The task on the IFN/ENIT database is single word recognition. Therefore no multi-gram language-model is used for recognition. In addition, the training vocabulary equals the possible words to be recognized, the OOV rate is 0%, so that a minimum

WER of 0% is theoretically possible. All result tables are adapted to the one's used for the ICDAR 2005 and 2007 competitions with the following sets (see Section 4.1):

Crossvalitation folds abcd
: The folds a-d are used as four-fold-crossvalidation folds as given at the ICDAR 2005 competition.

Evaluation fold abcd-e
: Evaluation is done on the fold e with sets a-d as training folds. This is comparable to evaluation fold e of the ICDAR 2005 competitions.

Additional results abcde-d/e
: Another evaluation of the ICDAR 2007 competition was done as training-on-the-testing-data result for which the systems - which were trained on sets a-e - were evaluated on set d and e.

For a basic experimental setup, the standard parameters of a speech recognition system are applied. The feature extraction is empirically optimized using PCA transformed motion features described in Section 3.3. The features are extracted on scale normalized images with a scale height of 16 pixels serving as the reference scale height for all following size measures. The sliding window has a width of seven slices with a width of one pixel each, and it is shifted by one pixel in writing direction. This results in a total number of $7 \times 16 \times 2 = 224$ features which are reduced using the PCA to a number of 30. These settings are verified by experimental evaluation as Figure 5.1 depicts. Though a sliding window with nine pixels width achieves slightly better results, a seven pixel width window is used. This is justified by the fact that the average character width at a scale-height of 16 pixel is seven pixels, thus a wider window would imply that more than half of a neighboring character would appear in the local character's context. For training and recognition a lexicon with MLE is used as described in [Dreuw & Jonas$^+$ 08] and Section 3.4.1.

A further overview of the impact of MLE is given later in this section on page 48.

White Space Modeling

Previous experiments have shown that the error rate increases the more PAWs a word contains. The visual inspection of the training-alignment of a first experiment with the basic setup displayed in Figure 5.2 explains this effect. The modeling of white spaces between compound words and PAWs is not yet sufficient and has a negative impact on the resolution of character models due to the implicit modeling of background within the models. Thus an explicit white space modeling as introduced in [Dreuw & Jonas$^+$ 08] is applied to the training-lexicon to improve the modeling of

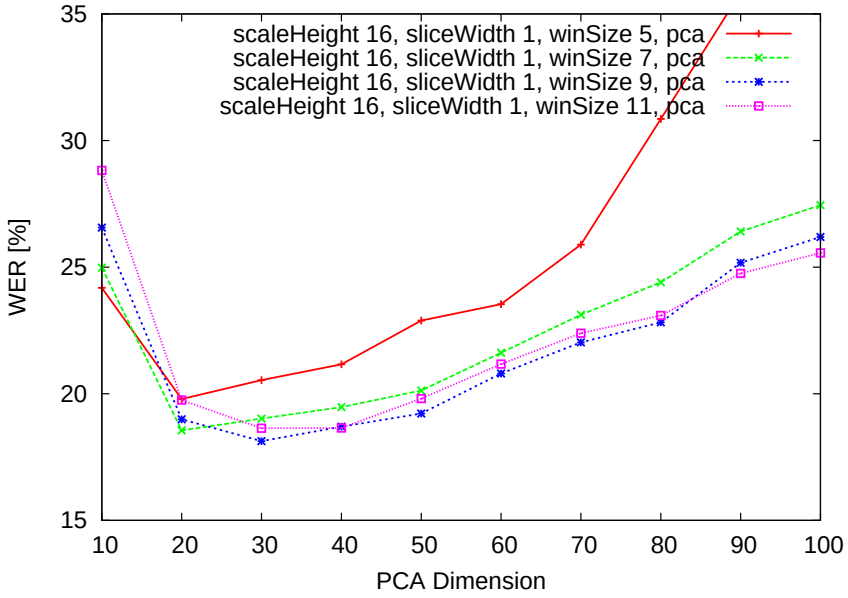


Figure 5.1: Comparison of different PCA dimensions and window sizes on the evaluation fold of the IFN/ENIT database

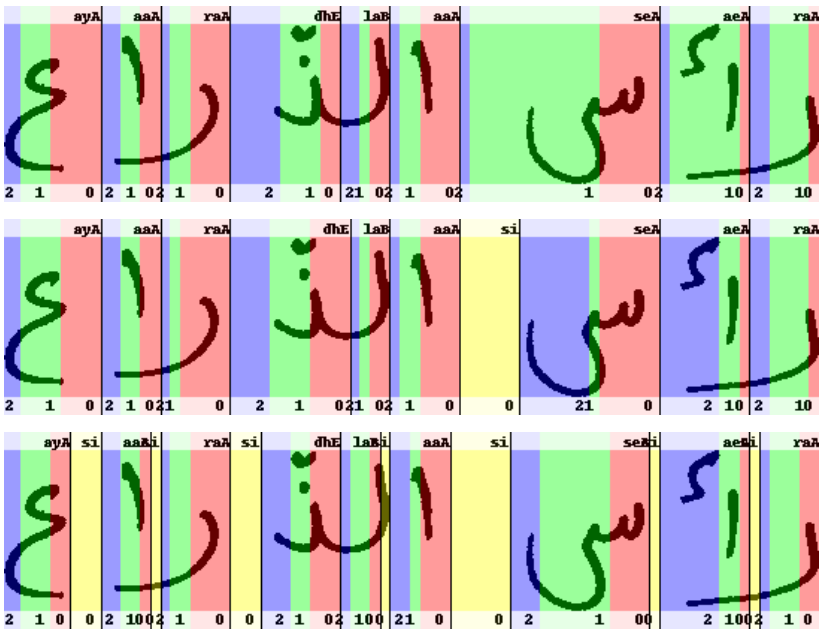
white spaces within a word. Different variants of white space modeling are tested according to Section 3.6.1.

The results displayed in Table 5.1 and Figure 5.3 as well as the visualisation in Figure 5.2 show improvements of the error rate and the training-alignment for the setups with explicit white space modeling as writing variants. The NB-lexicon without explicit white spaced performs worst on all folds since the white spaced between several characters are modeled implicitly by the surrounding characters. The alignment in Figure 5.2 shows that the BWWB lexicon fits best for both cursive (right images) and non-cursive writing (left images).

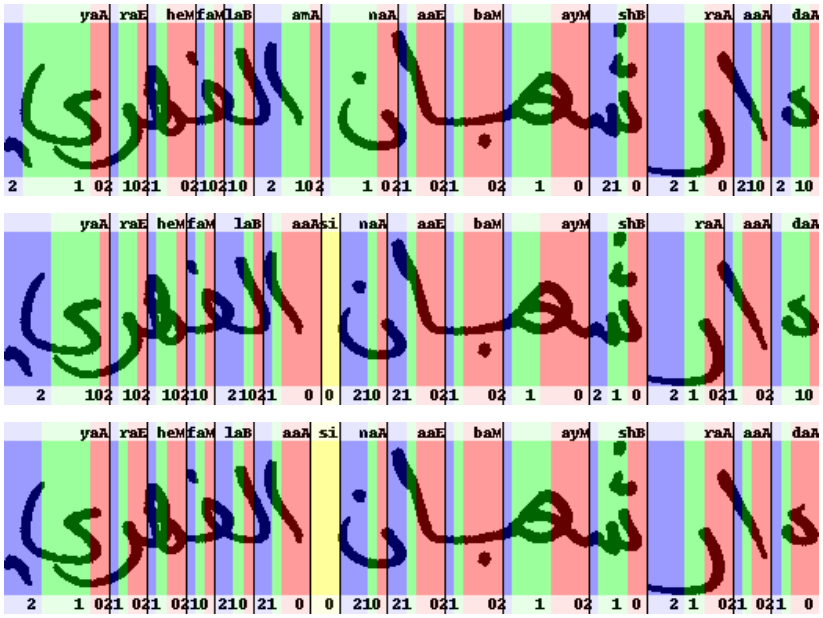
Consequently, the BWWB lexicon is retained for further experiments.

Supervised Writing Variants in Training

The first lexicon used for the basic setups contained several writing variants for each class (town identified by zip code) due to optional ligatures and diacritics. The choice of the correct writing variant was left to the system during training. Due to few training examples for some ligatures, those could not be learned and had to be eliminated from the character inventory of the lexicon and were replaced by one of their more common writing variants.



(a)



(b)

Figure 5.2: Alignment visualisation: two distinct words, one written cursive (a) and one non-cursive (b) trained with different white space lexica: First row NB lexicon, second row BWB lexicon, third row BWWB lexicon.

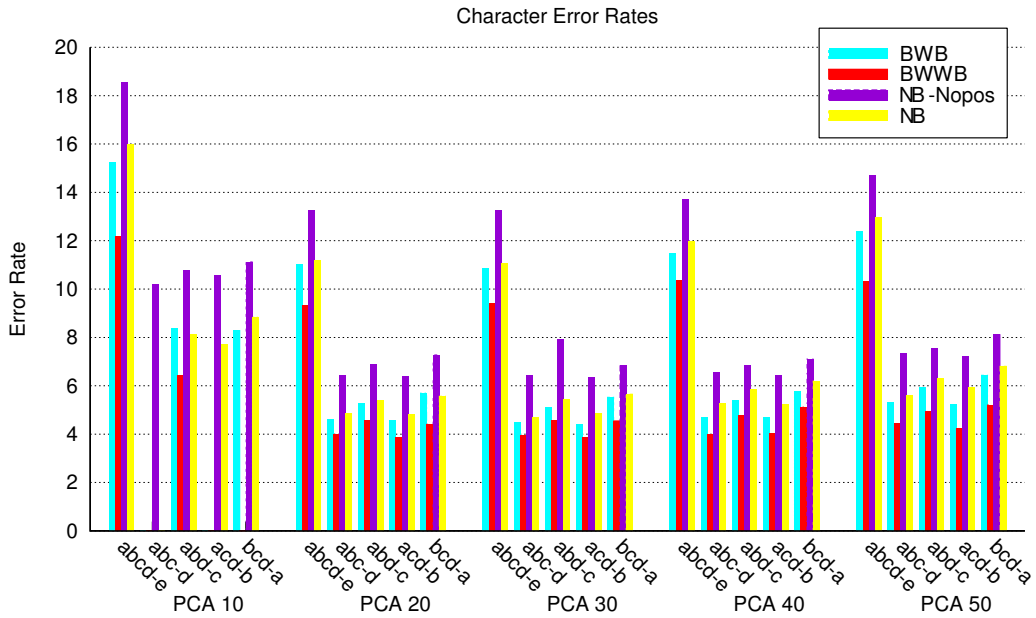


Figure 5.3: Comparison of different PCA dimensions and lexica with and without position information on the IFN/ENIT database

Table 5.1: Results: White space modeling with white space writing variants in the lexicon

Train	Test	WER [%]			CER [%]		
		NB	BWB	BWWB	NB	BWB	BWWB
abc	d	11.39	11.34	**10.78**	4.67	4.46	3.8
abd	c	12.57	11.95	**11.44**	5.43	5.09	4.37
acd	b	11.86	11.36	**10.88**	4.87	4.41	3.81
bcd	a	13.23	13.19	**11.86**	5.65	5.52	4.45
abcd	e	24.96	24.45	**23.54**	11.04	10.86	9.40

Table 5.2: Entries in the training lexicon and number of writing variants with and without supervised writing variants on the complete lexicon using BWWB white space modeling.

	BASE	SWV
#Entries	938	32493
#Writing Variants	2952	63500

Table 5.3: Results: unsupervised and supervised writing variants in training

Train	Test	WER [%]		CER [%]	
		BASE	SWV	BASE	SWV
abc	d	10.78	**7.80**	3.86	**3.12**
abd	c	11.44	**8.71**	4.37	**3.41**
acd	b	10.88	**7.84**	3.81	**2.87**
bcd	a	11.86	**8.663**	4.45	**3.52**
abcd	e	23.54	**16.82**	20.32	**7.52**
abcde	d	7.23	**3.44**	3.44	**1.32**
abcde	e	18.00	**10.09**	10.09	**3.81**

To compensate for this back draw and since the correct writing variant for each image is known in training, these annotations are used for a training lexicon with supervised writing variants (SWV). In the new lexicon each image is only associated with its own writing variant of the corresponding class and the additional variants for each white space model.

As Table 5.2 shows, the number of different annotations used increases significantly from the number of possible classes 938 to the number of images in the corpus 32493. The ratio of writing variants per image on the other hand drops from about three variants to less than two, which contain only variants of white spaces and not of the annotation. Thereby, less false variants can be chosen and an improvement of the visual model is expected. Table 5.3 shows a comparison of the unsupervised and SWV in training. The WER as well as the CER improve on all crossfolds. The biggest improvements can be seen on the validation fold where the WER drops by 5.5% and the CER by almost 13% absolute.

Accordingly, the training with SWV is able to improve the results and will be used for the subsequent experiments.

Table 5.4: Results: deslanted, UPV-preprocessed and non-preprocessed input data

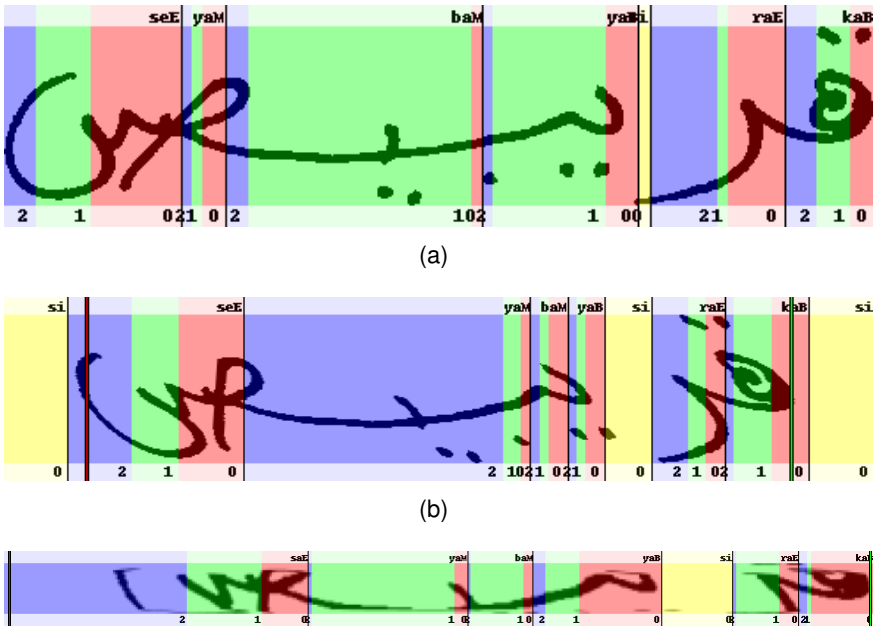
Train	Test	WER [%]			CER [%]		
		DESLANT	UPV	SWV	DESLANT	UPV	SWV
abc	d	11.89	32.74	**7.80**	4.15	9.58	**3.12**
abd	c	14.54	34.18	**8.71**	5.04	10.16	**3.41**
acd	b	13.46	33.14	**7.84**	4.53	9.70	**2.87**
bcd	a	14.46	35.96	**8.66**	4.98	10.90	**3.52**
abcd	e	22.71	53.39	**16.82**	8.67	16.02	**7.52**
abcde	d	6.33	19.27	**3.44**	2.14	5.69	**1.32**
abcde	e	14.02	40.46	**10.09**	5.29	12.37	**3.81**

Preprocessing

Most of the current Arabic handwriting recognizers make intensive use of preprocessing (see [Märgner & Abed 07] and [Märgner & Pechwitz$^+$ 05]). The system proposed here does not use any preprocessing for the input data. To test the impact of preprocessing on the system the preprocessing tools described in Section 3.2 are applied (UPV). Since the images in the IFN/ENIT database are binary images, the color normalization part of the preprocessing is not used. For a better evaluation of the complete preprocessing the deslanting (DESLANT) step is also used separately for a second training.

As the results in Table 5.4 show, the preprocessing does not improve the previous results. In contrary, the preprocessing significantly increases the error rates on all folds. A visual inspection of the training data might explain this (see Figure 5.4). Both the deslanting and the height normalization decrease the readability of the images. The desired effect of the deslanting - a better separation of characters - is even inverted such that characters that were beforehand side by side are now shifted beneath each other. The decrease of the height of ascenders and descenders performed by the size normalization as described in Section 3.2 removes additional important informations from the image, for example, dots that are located in the upper or lower part of the image. These dots are important since several characters of the Arabic alphabet differ not by shape but only by the position of dots, for example, Haa ح and Khaa خ.

The here applied preprocessing seems not to be suitable for Arabic HWR on the IFN/ENIT database and will therefore not be applied in subsequent experiments.



(a)

(b)

(c)

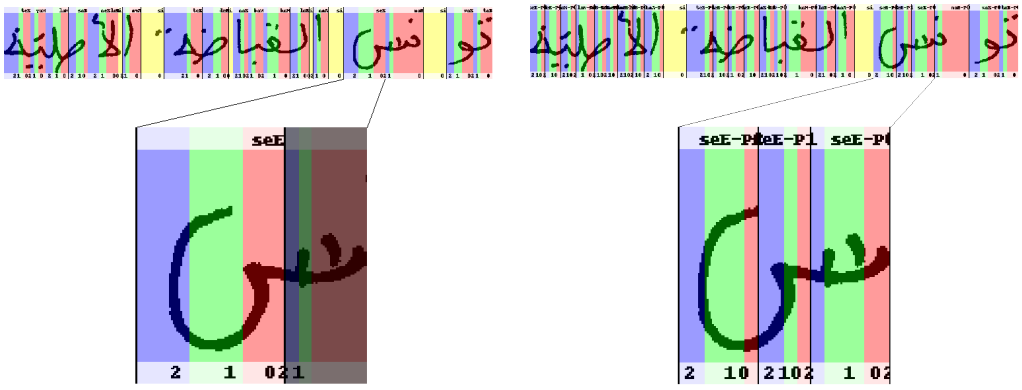
Figure 5.4: Alignment visualisation: preprocessing on the IFN/ENIT database: a) No preprocessing; b) Deslanted; c) UPV-preprocessing

Approximation of Model Length Estimation

For all previous experiments training lexica with MLE are used which models are complex. The MLE is calculated on the training data which poses the risk of overfitting on the character length. Thus, an approximation is introduced using a simple but flexible linear model with fixed number of states for all characters. During the implementation of the MLE lexicon the mean and median length of all characters are observed. The length of most characters is located within a range of three and six pixels length. Hence, a model with more than three different mixtures seems to be inappropriate. To avoid high loop penalties for longer characters a new topology for the model with three states and two repetitions per state is used (3-2, cf. Figure 3.2).

Additionally, the 3-2 model better compensates high length variations of the characters. This is important when grouping characters by their shape or by dropping the position information to create a reduced model set. For example, the mean length of the letter ب (Baa) in initial and medial position is six pixels but in final position is more than 17 pixels. If grouping these characters, a MLE would not suffice the length varieties within the character group.

The results in Table 5.5 show that the model with a 3-2 topology can achieve similar results as the MLE setup. For additional comparison the results of a model with three



(a) (b)

Figure 5.5: Comparison of training alignment using 3-1 (a) and MLE (b) topology on the IFN/ENIT database. The exposed character is aligned correctly only using the MLE setup, without MLE half of the character is aligned wrongly.

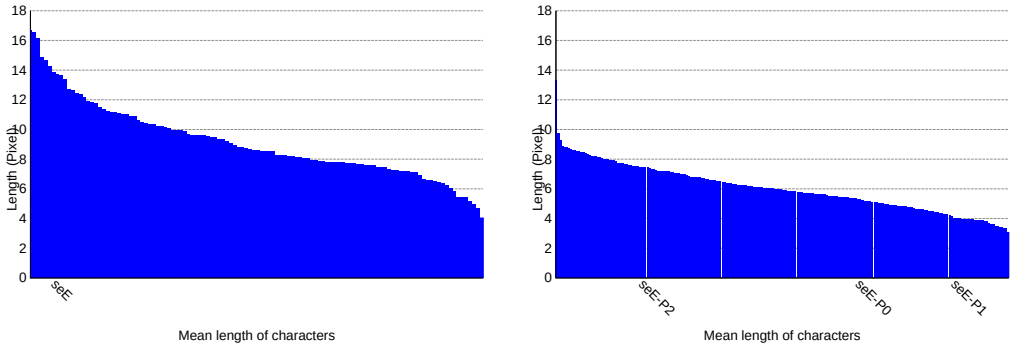
states and one repetition per state topology without MLE is also displayed in the table (3-1). The MLE setup improves the training alignment in comparison to the 3-1 setup as figure Figure 5.5. The spatial resolution is improved as well as depicted in Figure 5.6 where the mean lengths of the (pseudo) characters are more smooth using the MLE and by that a higher spatial resolution is achieved. The MLE setup outperforms the 3-2 topology setup on the evaluation set and when considering the character error rate (CER) which implies that MLE is an essential part and should be used for general evaluation purposes.

Still, the 3-2 model topology performs adequately compared to the MLE and can be used for experiments in which a MLE can not be implemented respectively its implementation does not make sense, for example, when using a reduced character set.

Reduction of Model Size

It is well known from the inspection of the Arabic handwriting that several characters have similar or identical shape. To exploit this fact for the reduction of the model size two different method are chosen:

No Position (nopos)
: The position-dependent characters can be reduced to their parental character, such that the position information is dropped and a common and position independent model is created. Thereby, the number of characters is reduced from 120 to 50 characters.



(a) (b)

Figure 5.6: Comparison of the mean length of (pseudo) characters using 3-1 (a) and MLE (b) topology on the IFN/ENIT database.

Table 5.5: Results: setup with a 3-2 respectively 3-1 topology without MLE, and the SWV setup with MLE

Train	Test	WER [%]			CER [%]		
		3-1	3-2	SWV	3-1	3-2	SWV
abc	d	11.00	8.06	**7.80**	3.66	3.12	**3.01**
abd	c	11.41	**8.63**	8.71	3.97	3.41	**3.39**
acd	b	11.13	**7.79**	7.84	3.65	2.87	**2.76**
bcd	a	12.01	**8.63**	8.66	4.31	**3.52**	3.76
abcd	e	22.41	17.77	**16.82**	8.33	7.52	**6.85**
abcde	d	6.40	**3.24**	3.44	2.05	**1.21**	1.32
abcde	e	11.27	**9.07**	10.09	4.14	**3.10**	3.81

Table 5.6: Results: reduced character set with shape-reduced and without position informations, and a character set with all informations

Train	Test	WER [%]			CER [%]		
		nopos	reduced	3-2	nopos	reduced	3-2
abc	d	10.99	12.98	**8.06**	4.18	4.41	**3.12**
abd	c	11.07	12.21	**8.63**	4.14	4.16	**3.41**
acd	b	10.86	12.76	**7.79**	3.90	4.12	**2.87**
bcd	a	11.29	13.25	**8.63**	4.31	4.54	**3.52**
abcd	e	21.43	26.17	**17.77**	8.31	9.96	**7.52**
abcde	d	9.87	8.60	**3.24**	3.82	2.93	**1.32**
abcde	e	21.31	18.47	**9.07**	8.40	6.89	**3.81**

Shape Reduction (reduced)

The number of characters can be reduced according to their shape and appearance as proposed by PARIS-V (see [Menasri 08], [Märgner & Abed 07]). For example, characters that only differ by a dot or a connection to another character on one side can be bundled to one new character. To keep both methods comparable the characters are reduced to a number of 49 characters with this method.

As illustrated in Table 5.6 none of the proposed methods can outperform the lexicon with all characters. This is explained by the information loss resulting from the character reduction However the model of the setup without position informations has only 15,594 densities which is less than half the number of densities than the full setup having 35,751 densities. But the error rate only drops by less than 4% on the evaluation set. The shape-reduced character set cannot outperform neither the nopos character set nor the complete character set. These results concur with the results achieved by the PARIS-V system at the ICDAR 2007 competition.

When improving runtime or memory usage of the system the approach of loosing position informations to produce a reduced character set can be of use. In other cases, the position and shape information of the characters seem crucial for the recognition result.

CART

For the modeling of triphone contexts, CART as described in Section 3.4.2 is widely used in speech recognition and is therefore tested in this work. Since yet no work has been done on CART and Arabic HWR, several possible questions for the splitting

of the CART nodes are considered. For the left or right context triphone-, state- and boundary questions are used, for example “is the right context a word boundary”. For informations about the central character several question set have been tested using the following possible questions:

- Singe-, double-, trippel dots and their relative positions
- Diacritics: Madda, Hamza, chadda
- Numbers
- Boundary of PAWs and relative position to white spaces
- Ligatures
- Visual appearance (e.g loops, multi piece)
- Stroke stop is needed when drawing the character
- Baseline and top-minuscule informations

Additionally, a second setup for a CART is used which uses a reduced character set without position informations. The position independent setup only uses questions about the characters possible positions, for example “is the character a possible final character”. To keep the CART setups comparable to the previous ones the number of splits is altered to achieve a similar overall number of densities. The number of mixtures without cart is 117 characters $\times$ 3 states = 351, using CART the number is enhanced to 400 with the full character set, the position independent character set is expanded from 50 characters $\times$ 3 states = 150 to 200.

Table 5.7 shows the results achieved with the full character set and a optimized question set (CART) in comparison to the previous achieved baseline result (3-2). The CART setup is not able to outperform the 3-2 setup, neither on the crossvalidation folds, nor on the evaluation fold. A similar effect is observed on the position independent experiments as Table 5.8 where the WER and CER increase on all folds also.

This effect can result from two possible reasons. At first, the chosen feature extractions uses a large window, which models the left and right context of the current character implicitly. Secondly, the usage of triphones may not be appropriate for handwriting as the variety of shape of a character only varies depending on the previous character but on the following. Except for the relative position to white spaces, the main difference within characters regarding the context is the vertical position of the connection to the left or right character respectively parts of the neighboring characters.

CART context modeling using the proposed questions does not improve the results on the IFN/ENIT database and will therefore not being used in further experiments.

Table 5.7: Results: CART experiments on the IFN/ENIT database

Train	Test	WER [%]		CER [%]	
		3-2	CART	3-2	CART
abc	d	**8.06**	10.96	**3.12**	4.04
abd	c	**8.63**	10.67	**3.41**	4.19
acd	b	**7.79**	10.34	**2.87**	3.61
bcd	a	**8.63**	11.47	**3.52**	4.63
abcd	e	**17.77**	22.54	**7.52**	9.17
abcde	d	**3.24**	6.24	**1.32**	2.36
abcde	e	**9.07**	13.36	**3.81**	5.27

Table 5.8: Results: CART experiments using the position question set on position independent characters on the IFN/ENIT database

Train	Test	WER [%]		CER [%]	
		nopos	CART	3-2	CART
abc	d	**10.99**	11.74	**4.18**	4.61
abd	c	**11.07**	13.40	**4.14**	5.02
acd	b	**10.86**	14.78	**3.90**	5.56
bcd	a	**11.29**	14.12	**4.31**	5.60
abcd	e	**21.43**	25.59	**8.31**	10.20

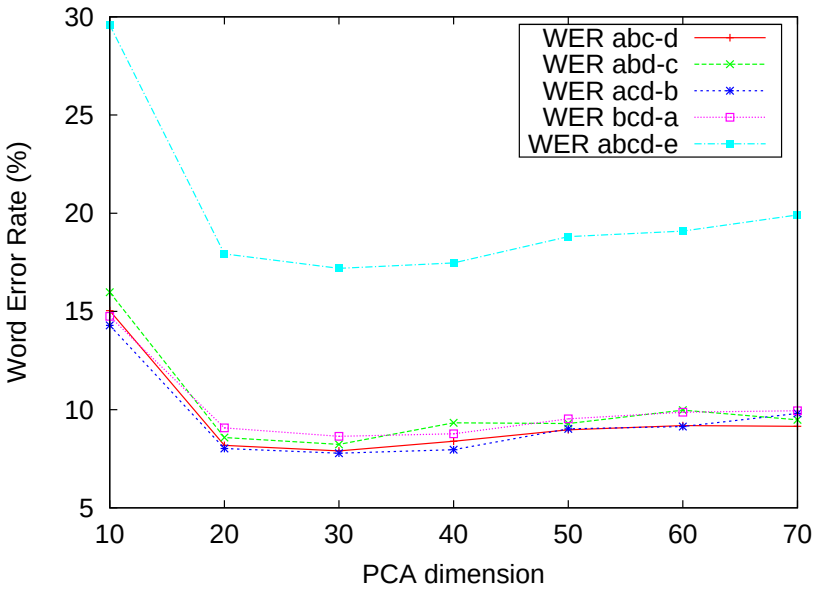


Figure 5.7: Comparison of different PCA dimensions for the Sobel feature set on the IFN/ENIT database

Feature Extraction

Previous results with handwriting recognition showed, that Sobel filtered images are often able to improve the results as illustrated by [Keysers & Deselaers+ 07]. Thus, a second feature extraction setup using the Sobel feature set as described in Section 3.3 is applied.

The motion feature set extracts two features (original + motion filtered) for each pixel in a 7×16 which makes a total of 224 features which are reduced with a PCA reduction to a dimension of 30. The Sobel feature set extracts five features (original + per pixel and by that has a total of 560. The dimension of the PCA reduction has been optimized empirically to 30 as well as displayed in Figure 5.7.

Table 5.9 shows that the Sobel feature set is able to outperform the motion feature set on three out of four crossvalidation folds. The WER on the evaluation fold also decreases about a half percent.

The Sobel features are able to improve the results on the cross validation as well as on the evaluation fold and can therefore be used in further experiments.

Table 5.9: Results: comparison of motion and Sobel feature sets

Train	Test	WER [%]		CER [%]	
		Motion	Sobel	Motion	Sobel
abc	d	8.06	**7.90**	3.12	**3.01**
abd	c	8.63	**8.23**	3.41	**3.31**
acd	b	7.79	**7.78**	2.87	**2.81**
bcd	a	**8.63**	8.64	**3.52**	3.60
abcd	e	17.77	**17.19**	7.52	**7.09**
abcde	d	3.24	**2.72**	1.32	**1.08**
abcde	e	9.07	**8.07**	3.81	**3.32**

Summary

The performance of the system in relation to current state of the art systems can be seen in Table 5.10. The table shows, that all other systems are outperformed.

5.3 IAM Database

The IAM database contains lines of handwritten text. The task to be performed on this database is a continuous word recognition task. Due to the varying number of words per line of text, the error rate is determined not only depending on substitutes words but also on inserted and deleted ones. Equation 5.1 shows the calculation of the WER. Character models are used for all experiments and they are mapped to words using a word lexicon.

5.3.1 Visual Modeling

This subsection discusses optimizations and improvements regarding the training of the visual model. All training and model parameters were optimized on the devel set and the results were verified on the evaluation set. For a detailed description of the sets see Section 4.2. The best training parameters obtained in the previous Section 5.2 but using the motion feature set are used for an initial setup since the input data of the IAM database are comparable to the IFN/ENIT database regarding scaling and resolution. Again, the reference resolution is 16 pixel height. An HMM model with three states and two repetitions per state is used in the baseline experiment. The features are extracted by a window sliding in writing direction left-to-right with a width of seven pixel and a shift of one pixel. The motion features as described in Section 3.3

Table 5.10: Summary of the results achieved on the IAM database in comparison with current state of the art systems.

	WER [%]	
	abc-d	abcd-e
Proposed Systems		
Baseline MLE	10.78	23.54
+ Supervised Writing Variants	**7.80**	**16.82**
+ 3-2 Topology	8.06	17.77
ICDAR 2005 results		
1. UOB	15.00	24.07
2. ARAB-IFN	12.06	25.31
3. ICRA (Microsoft)	11.05	34.26
...		
Other systems		
[Natarajan & Saleem$^+$ 08]	10.51	-
[Schambach & Rottland$^+$ 08]	-	18.11

are used and the feature dimension is reduced by a PCA to 30. Other dimensions are also tested but do not perform better as Figure 5.8 shows. In training, the mixtures are splitted in seven iterations which performed best as Figure 5.10 show. An LDA based feature reduction has also been tested, using the best training alignment achieved in this work as class labels but it was not able to outperform the results using the PCA for feature reduction as Figure 5.9 shows.

Preprocessing

The results of an initially performed baseline experiment depicted in Table 5.11 only achieve an WER of about 80%, which is obviously not satisfying. Examples of the corresponding training alignment in Figure 5.11 show that the alignment is sufficient for images with steady stroke thickness and a centered position of the text within the image but not for images with variations regarding these properties. Especially the varying heights of the baseline and top-minuscule lines in the images (see Figure 3.3 for details) as well as the unsteady stroke intensity and thickness combined with the gray bounding boxes around the connected components (as described in Section 4.1) seem to decrease the model quality and must be normalized during a preprocessing step. Hence, the preprocessing described in Section 3.2 is applied. In comparison to the alignment on the images without preprocessing, the alignment as indicated in Figure 5.11 is clearly improved. For example the word “MOVE” of the right image

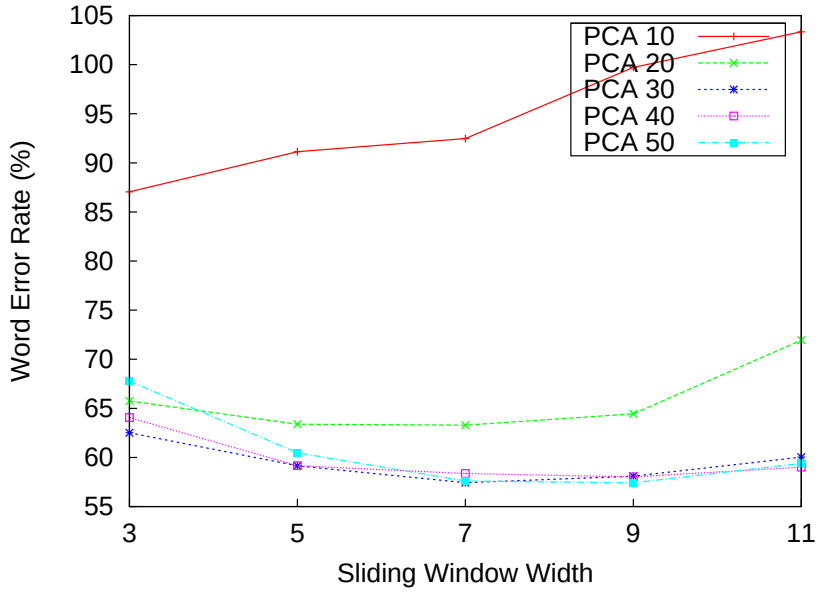


Figure 5.8: Comparison of the PCA dimension and sliding window width on the IAM devel set

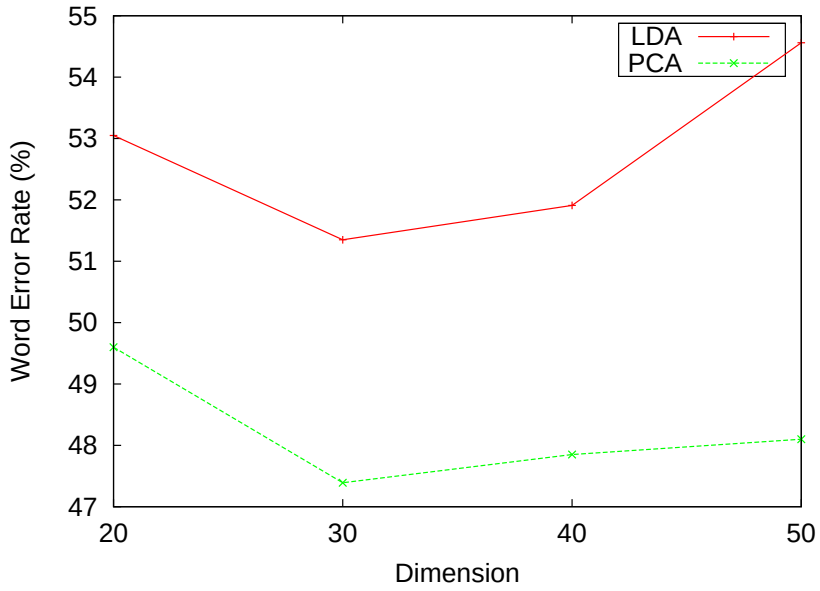


Figure 5.9: Comparison of PCA and LDA feature reduction with same setup on the IAM devel set

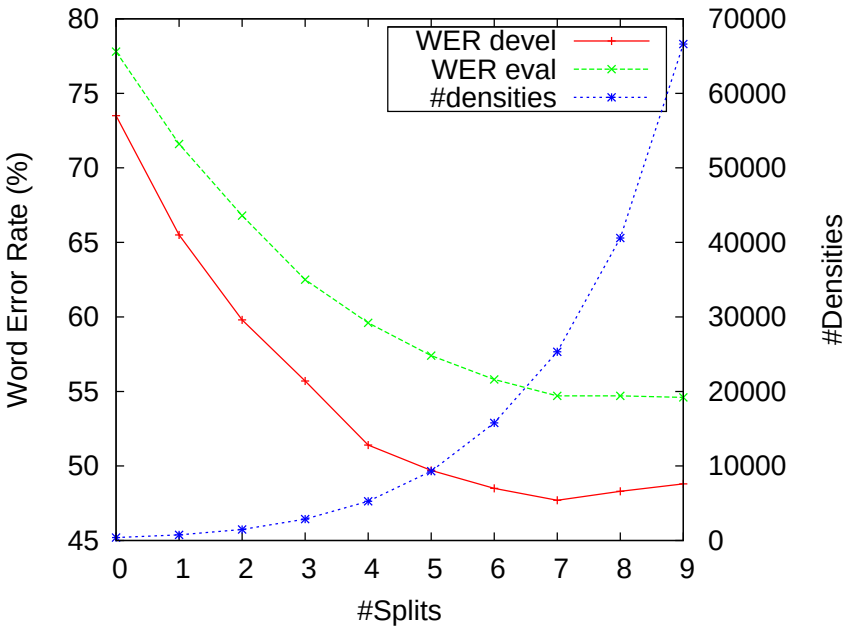


Figure 5.10: Comparison of the number of splits on the IAM database

is aligned erroneously without preprocessing: all characters are shifted to the left. The alignment on the preprocessed image show that all character models are aligned at least partially to the corresponding character in the image. The implicit modeling of the white spaces has improved too, fewer white spaces are aligned to character models. Though, the “A” of the preprocessed image on the right side is not correctly aligned.

Additionally, Table 5.11 shows that the WER on the preprocessed data decreases more than 30% absolute. Accordingly, a similar effect can be seen in the CER, which gained about 20%. This improvement recurs on the evaluation set.

Applying the preprocessing proposed by [Juan & Toselli$^+$ 01] the results improve strongly. Thus, only preprocessed data will be used for further experiments.

Model Length Estimation

Since the size normalization of the preprocessing changed the average width of all characters, it is necessary to recalculate the length of the model and hence the number of states used for the HMM. The calculation of the best model length is done using two approaches, the first one with a fixed number of states for all characters. The second approach uses MLE which has been used successfully on the IFN/ENIT

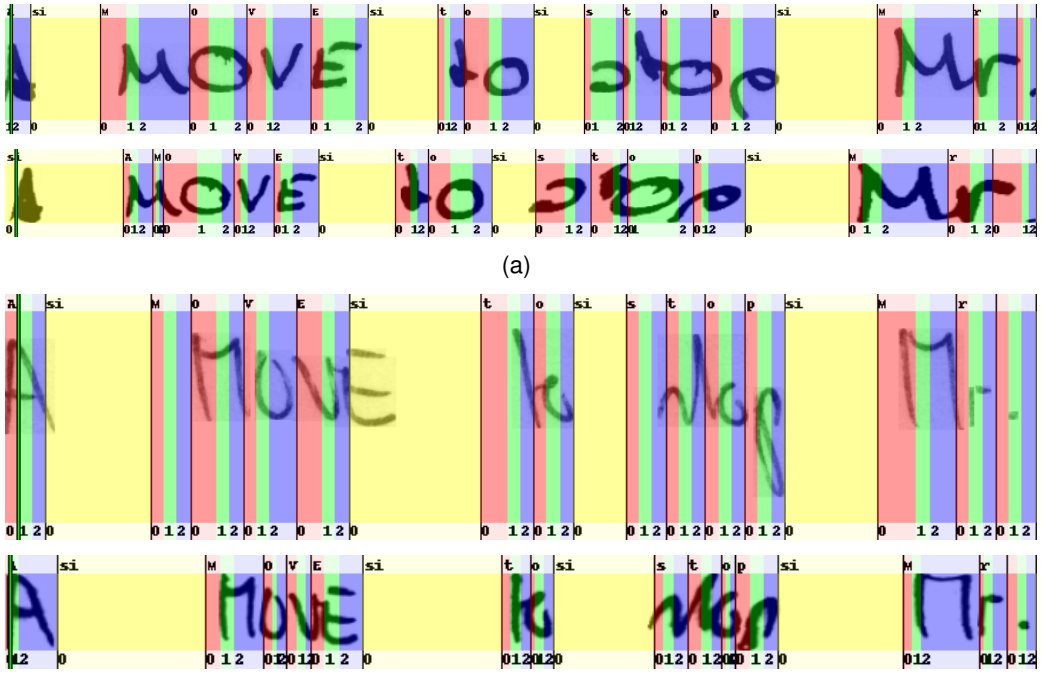


Figure 5.11: Comparison of training alignment of two example images a) and b) taken from the IAM database. Upper lines without preprocessing, lower lines with preprocessing.

Table 5.11: Results: Comparison of preprocessing

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
Baseline	81.07	34.28	6,569	1,328	13,321	83.60	37.82
Prepocessed	**57.59**	**15.08**	2,839	2,191	11,533	**65.26**	**19.27**

Table 5.12: Results: Comparison of model-width fitting by growing state number or MLE on the IAM database

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
3-2	57.59	15.08	2,839	2,191	11,533	65.26	19.27
5-2	**47.72**	10.32	3,112	1,526	9,236	**54.66**	13.76
MLE	54.91	**9.37**	5,890	422	9,533	62.43	**11.74**

database as Section 5.2 shows.

The mean length of all characters equals 12 pixels in comparison to about six pixels before preprocessing, calculated by counting all non-background columns on the input images and dividing their number by the complete number of characters at same scaling height. This result has to be seen as too high since the size normalization of the preprocessing does not resize all characters to identical width. For example, the width of narrow characters like “i” will be smaller than 12 pixel. Thus, to model short characters less than a total number of 12 states should perform best.

This corresponds to the results achieved with changing state numbers as depicted in Figure 5.12. The best results are achieved using an HMM with five states and two repetitions which best model characters with a width of about ten pixels and at least five pixels.

The length of the characters used for the MLE are extracted on the best results achieved with a fixed number of states. But since the preprocessing used on the input images includes a size normalization step and by that alls characters should be resized to similar width and height, only a small improvement is to be expected. The MLE does not use state repetitions.

Table 5.12 shows the results of the estimated model length in comparison to the baseline experiment (3-2). The results of the estimation with fixed model length of five states and two repetitions (5-2) outperforms the other tested models. The 5-2 setup improves the results compared to the baseline by about 10% WER and almost 6% CER on the devel set. The experiment using MLE does not achieve a similar result but still improves the WER about 3%. It even outperforms the 5-2 setup regarding the CER. The relatively worse WER can be explained by the high number of deletions which implies that short words are deleted but long words are modeled better which also explains the low CER.

An HMM with five states and two repetitions each performs best and will therefore be used for further experiments.

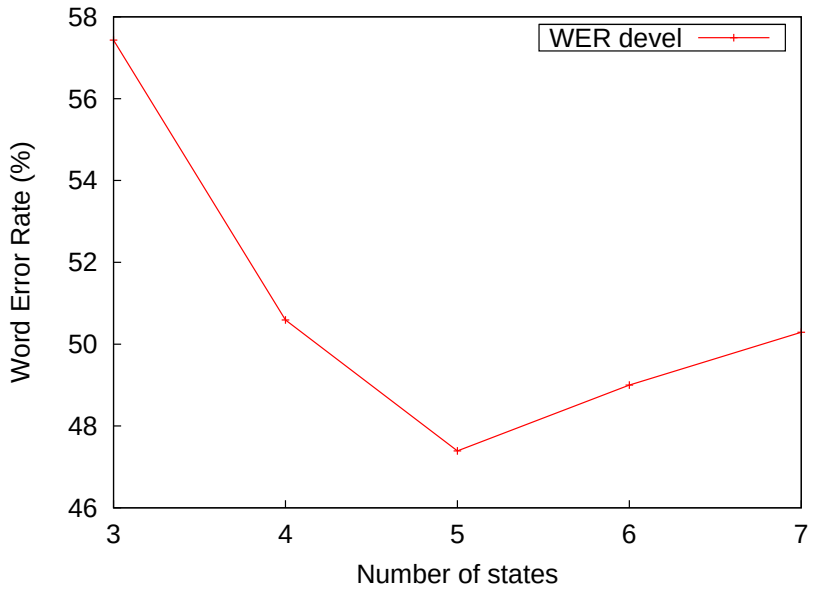


Figure 5.12: Comparison of growing state numbers of the HMM on the IAM database

Feature Extraction

For comparison with the results of the feature extraction in Section 5.2, the Sobel features are extracted on the IAM database as well. The dimension of the PCA reduction was optimized again and set to 50. Table 5.13 shows in contrary to the improvements made on the IFN/ENIT database, the Sobel features are not able to outperform the motion features but achieve similar results. The WER increases about 1% when using the Sobel features and the CER increases as well on the devel and on the eval set.

Sobel features do not outperform the motion features but increase the number of features by a factor of almost two. Therefore, the motion features are used in further experiments.

CART

As in Section 5.2, context modeling using CART is applied to the IAM database as well. The same context questions are used as for Arabic handwriting but different questions are chosen for the central character:

- Upper- and Lowercase characters, baseline and top-minuscule line informations
- Numbers

Table 5.13: Results: Comparison of different features

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
motion	**47.72**	**10.32**	3112	1526	9236	**54.66**	**13.76**
Sobel+Abs	48.80	10.41	2827	1692	9515	55.29	13.82

Table 5.14: Results: Comparison of CART experiments on the IAM database

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
No-CART	47.72	10.32	3112	1526	9236	54.66	13.76
Base-CART	60.36	14.71	1562	4230	11847	69.51	17.99

- Punctuation-, quotations marks and other symbols

The number of mixtures without cart is 80 characters $\times$ 5 states = 400, using CART the number is enhanced to 1000.

The results in Table 5.14 illustrate that CART is not able to improve the results of this experiment, also. In contrary, the error rates increase by almost 15%. The explanation for this is analogous to the Arabic handwriting: The triphone context and the feature extraction seem not to fit the provided data. In addition, the Latin handwriting differs in style and connectivity from the Arabic handwriting. For example, only four characters (o, r, v and w) do not have a connector at the baseline and by that alter the style of the succeeding character. Additionally, even if written cursive, characters seldom overlap due to the deslanting step in the preprocessing. Thus, only minimal variations depending on the context are to be expected and by that the context depending modeling might not be able to improve the results at all.

CART does not improve the results on this experiment with Latin handwriting and will therefore not being used in further experiments.

5.3.2 Language Models and Recognition

In contrary to the previous subsection, which dealt with the improvements of the visual models, this section deals with the modeling related to the recognition step. This includes not only experiments using LMs and lexica trained on additional data, but

also the subtext structure the recognition is performed on, for example, recognizing syllables instead of complete words.

Language Models

As reported by [Pitrelly & Roy 03] as well as by [Marti & Bunke 02b], the impact of LMs on continuous HWR should not be underestimated. The lexica and LMs used for most experiments were gathered as described by [Bertolami & Bunke 08] in order to reproduce the results presented in their paper. The LM is compiled as described in Section 3.5 using the LOB, Brown and Wellington (LBW) corpora (for a description see Section 4.3). The sentences from the corpora were concatenated and then splitted every ten words to simulate line writing. An approach without this simulation was also applied but performed worse. The perplexities of the LMs regarding the testing data are illustrated in Figure 5.13. The perplexity of the LMs decreases significantly from unigram to bi- and trigram. The decrease from trigram to higher n-grams is much smaller, which indicates that the used training data does not match the test data, which is not probable since the LOB corpus is the underlying corpus of the test data, or too little training data. With more training data, the perplexity should decrease the more context is taken into account. The constantly higher perplexity of the LBW LMs compared to the LMs generated on the training data of the IAM database (Train-LM) is induced by the higher amount of data used for the generation of the LBW LMs.

As Table 5.15 shows, the OOV rate of the training lexicon on the devel and eval data is about 15% which is high compared to the OOV rate of 6.3% proposed by [Bertolami & Bunke 08]. Hence, different lexica containing the 10 to 50 thousand (10k,...,50k) most frequent words from the LBW corpora are created to compensate for the high OOV rates as Table 5.15 illustrates. Since the LOB corpus is the underlying corpus of the IAM database (see Section 4.2), the lines occurring in the devel and eval set have to be removed before using it for the LM generation to avoid training of the LM on the testing data.

Figure 5.14 depicts the impact of the different n-grams and the number of words in the lexica. As supposed, both the WER decrease with an ascending number of n-grams. Analogously to the perplexity shown in Figure 5.13 WER and CER stabilize with the trigram LM. The differences between the WER with the same n-gram but different lexica can be explained by the OOV-rate of each lexicon. For example, the difference between the OOV rates of the 10k and 20k lexicon is about 4% which equals the difference between the WERs the two lexica achieve.

As Table 5.17 and Figure 5.14 show, the best WER is reached using a trigram LM and the 50k lexicon with 33.84% WER on the devel and 39.94% on the eval set. In comparison to the training lexicon and LM, the WER is topped by more

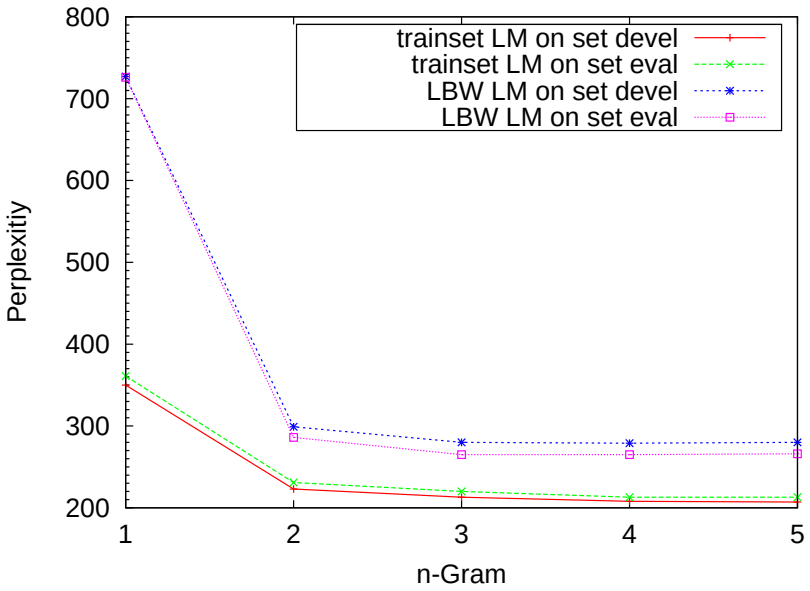


Figure 5.13: Perplexity of different LMs build on the training data and the combined LBW corpora

than 10% absolute on the devel set. The CER improves as well. Additionally, the results achieved with a 20k lexicon are depicted in the table for comparison with [Bertolami & Bunke 08] who used a 20k most frequent word lexicon from the same corpora and reached a WER of 35%.

An inspection of the results obtained reveals that the annotation of the database and the corpora used for the LMs are not consistently tokenized. Table 5.16 shows an overview of the most frequent confusion pairs. Several of the errors on the original data originate from bad tokenization regarding the suffixes "`'t`" and "`'s`". For example the tokenizations "[`tonight's`]" as well as "[`Something`] [`'s`]" can be found in the annotation of the database. Thus, the retokenization of the databases' annotation as well as the text corpora are necessary. Additionally, a mapping of ambiguous writing variants, for example "`Mr`" and "`Mr.`" was applied so that all writing variants are represented equally in the LM. The retokenization of the corpora and annotation further improves the results in Table 5.17. Both the WER and CER are improved on the devel set as well as on the eval set. The confusion pairs displayed in Table 5.16 are now free of tokenization errors. When comparing the results of the original with the retokenized annotation in Table 5.17, the number of substitutions stays almost the same while the number of deletions decreases about 10% relatively. This is an effect of the retokenization because when recognizing, for example, "[`isn't`]" instead of

Table 5.15: Comparison of OOV rates of the lexica on devel und eval set

Corpus	Lexicon size	OOV-Rate [%]	
		devel	Eval
trainset	7,764	15.45	15.03
	10k	10.14	10.06
	20k	6.96	6.54
LBW	30k	5.42	4.92
	40k	4.65	4.01
	50k	4.01	3.47

“[isn] [’t]”, one substitution and one deletion is counted. After the retokenization, these deletions are avoided. After the retokenization, most errors are substituted punctuation marks, which is most likely due to the low resolution of the input images, and a mix-up of upper- and lowercase characters.

To compensate for the latter error, a lowercase LM is created based on the retokenized corpora and all words in the lexicon are mapped to their lowercase equivalent. Thus, the choice between upper- and lowercase is made only based on the visual model. Nonetheless, a completely lowercase LM poses a risk, since a lot of information is lost and the quality of the visual models might not be sufficient to recognize the cases adequately. The information loss is especially important for punctuations, since the most punctuation marks are followed by an uppercase character. The results achieved with the lowercase LM are also displayed in Table 5.17 but their WER increases about 13% absolute compared to a true case LM. When using a lowercase LM the confusion matrix shows that more upper- and lowercase characters are mixed up.

The experiments with different LMs show that a proper tokenization and generation of LMs is necessary for good recognition results.

Character Recognition

A problem when recognizing continuous text is the occurrence of words not covered by the recognition lexicon. Those words cannot be recognized since they are not known neither by the recognition lexicon nor by the LM. One possibility to recognize these words correctly is a character based recognition, where the lexicon contains only the characters that have been seen in the training and an explicit white space model. The white space model is used as boundary in the reconstruction of the words. A character LM is trained on the training data and on the LBW corpora with increas-

Table 5.16: Top10 confusion pairs with original and retokenized annotation an LMs on the IAM database

Top	Original			Retokenized		
	Reference	Recognized	#	Reference	Recognized	#
1	;	,	10	;	,	11
2	’t	didn’t	8	,	.	10
3	,	.	8	“	.	8
4	“	.	7	.	,	6
5	’t	don’t	7	Broughtons	Broughton	6
6	.	,	7	his	this	6
7	was	has	7	was	has	6
8	’t	isn’t	6	He	the	5
9	his	this	6	”	in	4
10	He	the	5	...	.	4

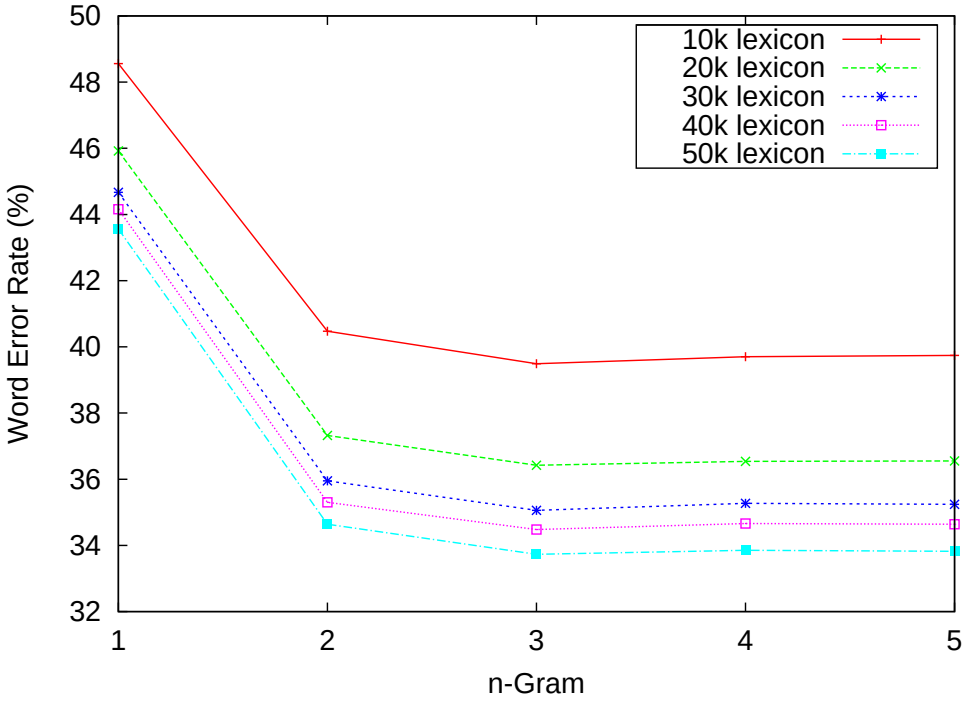


Figure 5.14: Results for different lexica sizes using the LBW corpus and corresponding lexica when recognizing on the devel set

Table 5.17: Results: Comparison of word LMs and lexica

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
Train-LM & Lex	47.72	10.32	3112	1526	9236	54.66	13.76
LBW LM (3gram)							
+ 20k most Lex	36.24	9.05	3138	838	6671	42.13	11.13
+ retokenized LM	34.64	8.92	2790	1080	6690	41.45	10.97
+ lowercase LM	41.37	9.84	3285	867	7882	47.41	12.02
+ 50k most Lex	33.84	8.61	3354	623	6159	39.94	11.95
+ retokenized LM	**31.92**	**8.44**	2942	800	6188	**38.98**	**11.79**
+ lowercase LM	45.17	9.83	3823	586	8770	51.93	13.52

ing context length. The best WER with word recognition was achieved with a trigram. A word in the training data is composed of five characters in average. Therefore, a character recognition with a 15-gram character LM is supposed to achieve the best results. But as Figure 5.15 shows, the perplexity of the LBW LM on the devel set decreases at first but increases after reaching the seven-gram. The perplexity basically measures the number of possible tokens in the LM within a certain context. A perplexity of six basically means that at every position only six out of 80 characters are likely. To compare the achieved results with the word based recognition the seven-gram LM will be applied.

Since a character recognition is able to make more errors in spelling than a word based system, a spell checker [1] is applied to the recognized text in a post-processing step. The results of the Train LM and the LBW corpus without and with spell checking are provided in Table 5.18. When using the Train LM, the WER is decreased about 4% in comparison to the word recognition. But since the OOV rate of the training lexicon is about 15%, a total loss of about 10% WER is calculated. When comparing the results achieved by a word based recognition using the LBW LM, the difference regarding the OOV rate is only a loss of about 5%. The results also show, that the spell checker can only improve the results created using the Train LM but not the results achieved with the LBW LM. This implies a good modeling of the testing data by the visual model and the character LBW LM.

Since the character recognition only uses a seven-gram, it might be able to perform adequately compared to word base recognition when using an LM trained with more

[1] ISPELL with standard settings; http://www.gnu.org/software/ispell/ispell.html

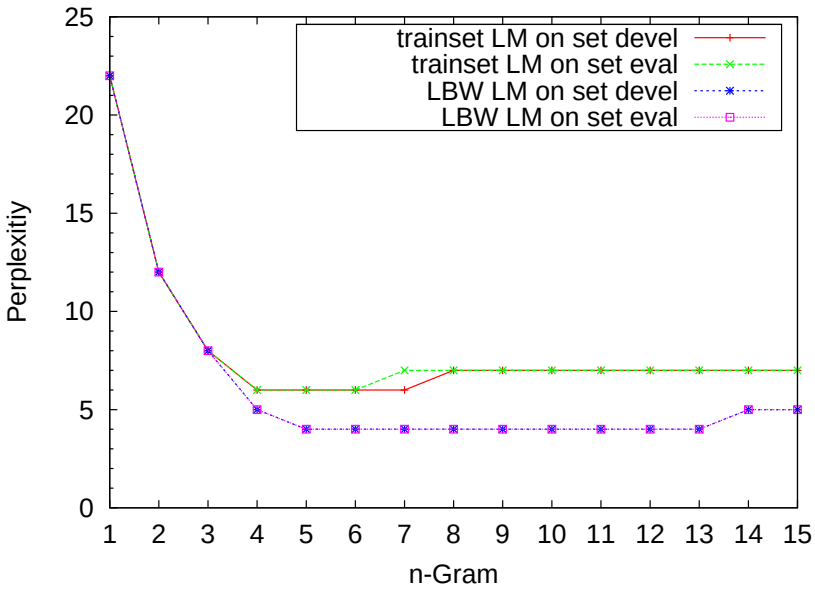


Figure 5.15: Perplexity of different character LMs on the training data and the LBW corpora

data.

Syllable Recognition

Another possibility to reduce the number of unknown words in the lexicon is syllable recognition. This method does not erase all unknown words, but the number of possible syllables is much smaller than the number of possible words. For example, to model all words of the 20k most frequent word lexicon extracted from the LBW corpora, only 13 thousand syllables are needed. In addition, syllables possess a level of information that is not as low as characters' information level, several syllables themselves form a valid word.

Therefore, the OOV rate drops substantially when using the 20k most frequent syllables instead of the 20k most frequent words. The OOV rates of the syllable lexica displayed in Table 5.19 are much smaller than the OOV rates of the word recognition.

In order to perform a syllable recognition, syllable lexica and LMs are created according to Section 3.7. Similar to the character recognition, an explicit white space model is used as indication of word boundaries for the word reconstruction. Also, a spell checker is applied in a post-processing step to improve the results.

Figure 5.16 depicts the perplexities of the syllable LMs. Again, it indicates that

Table 5.18: Results: Character recognition with character LMs and lexica

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
Word Recognizer	31.92	8.44	2942	800	6188	38.98	11.79
Train-LM	43.78	21.90	2175	1097	10212	52.93	28.47
+ Spell checker	42.89	22.94	2175	1097	9899	51.70	29.67
LBW LM (7-gram)	**35.42**	**16.28**	3631	301	6637	41.49	**21.37**
+ Spell checker	35.64	17.09	3631	301	6611	**41.39**	22.07

Table 5.19: Comparison of OOV-rates of the syllable lexica on devel und eval set

Corpus	Lexicon size	OOV-Rate [%]	
		devel	Eval
trainset	5,726	4.60	4.41
LBW	10k	2.33	1.78
	20k	1.44	0.92
	30k	0.93	0.59
	40k	0.76	0.50
	50k	0.63	0.43

the data used for training is not sufficient but will be used for comparison reasons. Since each word exists of approximately two syllables, a five-gram LM is chosen to reproduce results obtained using a trigram word LM.

When comparing syllable to word and character recognition the information level of syllables is higher compared to characters but lower compared to word recognition. On the other hand is the OOV rate of word recognition higher than the one of syllable recognition, character recognition has no OOV words. Therefore, the syllable recognition is expected to perform better than the character recognition but worse than the word recognition.

Table 5.20 shows the results obtained using the training and LBW LM and a 20k lexicon. None of the results achieved with syllable recognition is able to outperform a word or character recognition with the same or even less training data. When comparing the Train-LM with the LBW LM the WER decreases while the CER increases. This might be caused by sub-word constraints given by the syllable language model.

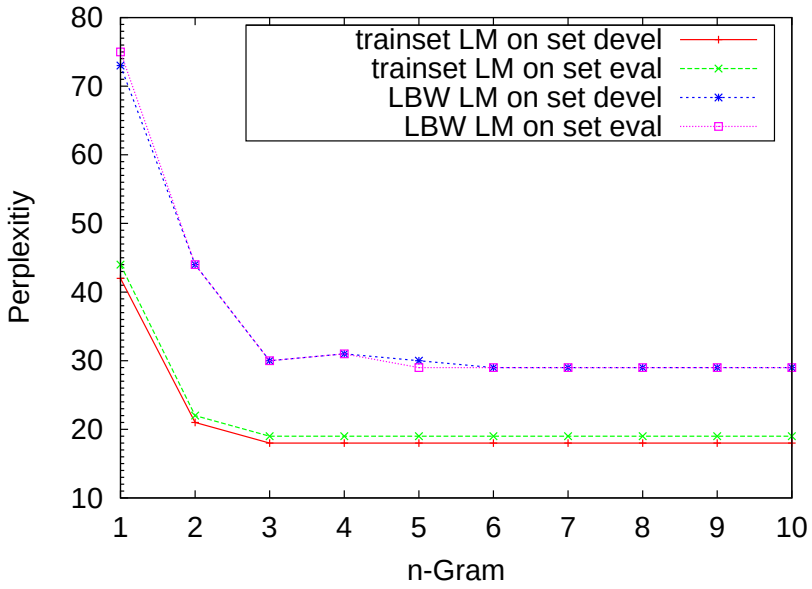


Figure 5.16: s
Perplexity of different syllable LMs on the training data and the LBW corpora

Table 5.20: Results: Syllable recognition with syllable LMs and lexica

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
Word Recognizer	31.92	8.44	2942	800	6188	38.98	11.79
Train LM	48.00	**24.44**	3702	567	9102	52.49	**28.98**
+ Spell checker	47.89	25.35	3706	564	9036	52.23	29.64
LBW	46.81	24.47	2960	972	9417	52.40	0 30.47
+ Spell checker	**45.96**	25.74	2961	969	0250	**51.74**	29.41

5.3.3 Discriminative Training

[Dreuw & Heigold+ 09] proposed a MMI based discriminative training and evaluated it on the IFN/ENIT database. They were able to improve the results by about 30% relative. The same algorithm, also described in detail in Section 3.4.3, is applied to the best models generated in this work using the LBW word lm and the character visual models with five states and two repetitions. For the training-recognition, a word unigram is chosen. The discriminative training is then performed with a maximum of 50 RProp iterations, the according WERs are illustrated in Figure 5.17. For the final recognition, the model generated in iteration 42 is chosen to avoid overfitting, since the WER on the devel set slightly increases in the following iterations, even if the CER still decreases.

The result achieved is displayed in Table 5.22 using a 20k and a 50k lexicon. The results improve significantly to less than 30% on the devel set and about 35% on the evaluation set using a 20k lexicon. With this results we are able to compete with the current state of the art by [Bertolami & Bunke 08] who achieved also 35% WER on the evaluation set.

An inspection of the confusion pairs in Table 5.21 shows, that most errors based on regular characters are eliminated. In contrast, errors regarding punctuation and quotation marks are increased. This can be explained with a visualization of the training alignment in Figure 5.18. The punctuation marks, for example, the dot after "Mr" do now implicitly model the context as a part of the previous character is aligned to the dot. This might decrease the quality of the punctuation mark models. On the other hand, the visualization revealed that the modeling of white spaces further improved as less white space is modeled implicitly by the surrounding characters, for example after the "to" in Figure 5.18(a).

The discriminative training is able to further improve the results by especially improving the implicit modeling of white spaces and shall therefore be used in ongoing experiments.

Summary

The performance of the system in relation to current state of the art systems can be seen in Table 5.23. The table shows, that all other single word recognition systems are outperformed. The proposed system can only be outperformed by the ensemble system proposed by [Bertolami & Bunke 08].

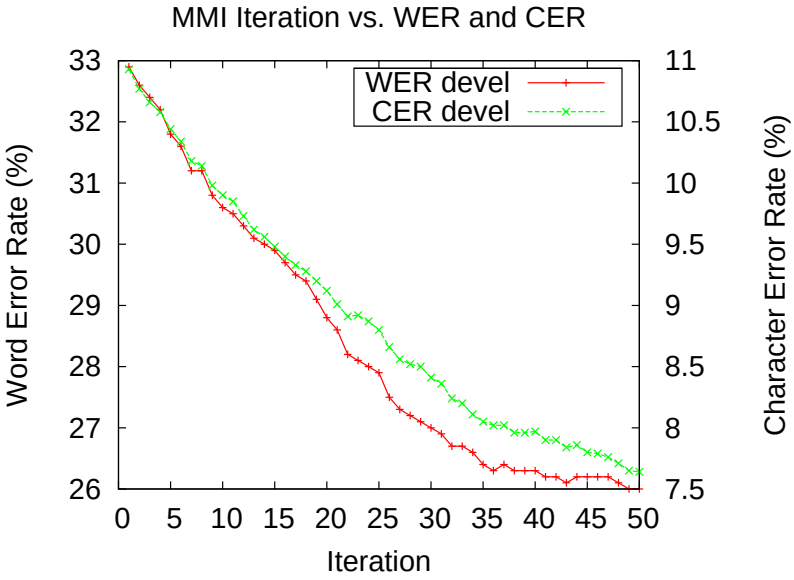
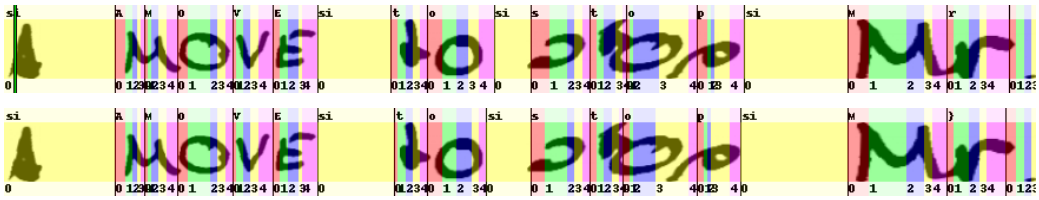
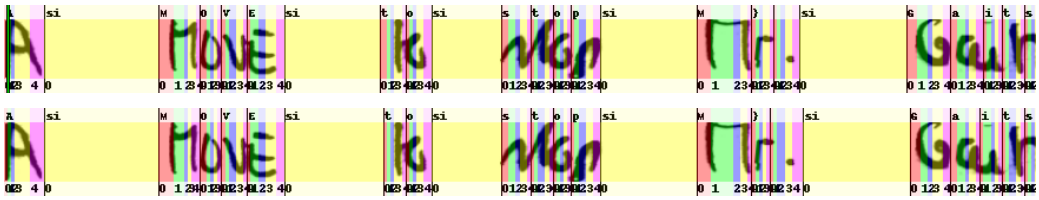


Figure 5.17: WER and CER using different RProp iterations of the discriminative training



(a)



(b)

Figure 5.18: Comparison of training alignment of two example images a) and b) taken from the IAM database. Upper lines without 5-2 baseline setup, lower lines with additional discriminative training.

Table 5.21: ableop10 confusion pairs with retokenized LM and discriminative training on the IAM database

Top	Reference	Recognized	#
1	,	.	27
2	.	,	21
3	;	,	16
4	”	’	8
5	”	,	8
6	’t	not	6
7	isn	is	5
8	was	has	5
9	“	’s	4
10	”	,	4

Table 5.22: Results: Recognition using discriminative trained models

Experiment	Devel		Eval				
	Words	Chars	Words				Chars
	WER [%]	CER [%]	Del	Ins	Sub	WER [%]	CER [%]
No-DT	31.92	8.44	2942	800	6188	38.98	11.79
With-DT							
+ 20k most Lex	29.40	7.18	1268	1678	6052	35.32	**11.18**
+ 50k most Lex	**26.19**	**7.90**	1401	1304	5580	**32.52**	12.40

Table 5.23: Summary of the results achieved on the IAM database in comparison with current state of the art systems.

	WER [%]	
	Devel	Eval
Proposed Systems		
Baseline	81.07	83.60
+ Prepocessed	57.59	65.26
+ LBW LM % Lexicon	34.64	41.45
+ character recognition	35.42	41.49
+ discriminative training	**29.40**	**35.32**
Single Word Recognizer		
[Bunke & Bengio$^+$ 04]	-	≈ 42
[Bertolami & Bunke 08]	30.98	35.52
[Natarajan & Saleem$^+$ 08]	-	*40.01*
[Romero & Alabau$^+$ 07]	*30.6*	-
Ensemble systems		
[Bertolami & Bunke 08]	*26.85*	*32.83*

Chapter 6

Conclusion

In this work, a script independent HWR system was presented and evaluated on Arabic and Latin handwriting. Several methods for improvement of language-, context- and visual models were tested. In contrary to other systems which use advanced preprocessing and feature extraction, it was shown that a basic preprocessing for position and size normalization as well as appearance based features on low resolution images are sufficient for HWR.

For the Arabic single word recognition task different lexica, preprocessing and context modeling were tested. On the Latin handwriting a continuous word recognition task was performed and improvements were made on the preprocessing, lexica, context- and language models mainly. Additionally, sub-word recognition was evaluated.

Experiments on single word recognition using lexica with explicit white spaces between compound words or sub-words revealed that the explicit white spaces within words improve the recognition results as a better segmentation of the sub-words is achieved. Furthermore, lexica using word wise writing variants were compared to lexica using supervised writing variants for each input image. The usage of supervised writing variants can further improve the results achieved since the automatic mapping of writing variants to an input image is not as reliable as the manual annotation of the images.

The impact of MLE was investigated in comparison to more simple but flexible models. As an alternative to the complex MLE models an approximation was found using a fixed number of states with state repetitions which achieves similar error rates on the cross validation folds. The model was adjusted resulting in a maximum spatial resolution of the characters with minimal length and with the lowest TDPs for characters with average length. This simpler model can be used if the length of the model cannot be estimated reliably.

For further improvements of the visual models on the Arabic database, possibilities of simplification of the models were compared. Two methods were proposed, one using a shape based reduction of the models such that similar characters are represented by one model. The second method uses one model per character instead of four by not using the position information. Both methods are not able to outper-

form a system trained with the full information of shape and position of the character. Nonetheless, the models trained with a reduced number of characters are much smaller and thus can be used when optimizing the system for a faster recognition.

The best error rates on the IFN/ENIT database were obtained using a model with MLE and supervised writing variants. These results outperform most of the published results and the proposed methods can therefore clearly be seen as state of the art.

On the IAM database the visual models were improved in a similar manner as on the IFN/ENIT database. The preprocessing applied to the IAM database was able to improve the results significantly. Most important for this task is the size normalization as the input data was not normalized in height or text position.

A length estimation of the characters was performed on the preprocessed data using fixed and variable number of states per character. The MLE was not able to outperform an estimation with a fixed number of states which can be explained by the size normalization of the preprocessing.

Most important for continuous word recognition was the creation of LMs and lexica. Different LMs were tested, using the training data and three additional text corpora. Results using an LM trained on the training data performed badly due to a high OOV rate. The error rates were improved by generating an LM and a corresponding most frequent word lexicon using additional data. Thereby, the tokenization of the database and the text corpora was modified to a consistent format. A lowercase and reduced LM along with a true case LM were tested. The best performance was reached using the true case LM with the retokenized database and corpora.

Moreover, the word recognition was compared to character and syllable recognition. Therefore, character and syllable LMs and lexica were generated using the improvements made in the word recognition. Both sub-word recognitions were not able to outperform the word recognition using the additional text corpora. But when dealing with texts with high OOV rates in the word recognition, the character recognition performed well as it has an OOV rate of 0%.

For further improvements of the visual models, a discriminative training was applied to the IAM database. Visual inspection showed, that the models were improved especially regarding the surrounding white spaces. After applying the discriminative training, the white space was less often modeled implicitly which decreased the error rate. On the other hand, more errors were made regarding the punctuation mark as more context of previous characters was modeled implicitly.

The best results were achieved using discriminative training and and optimized LM on the IAM database and can slightly outperform the currently best results reported by [Bertolami & Bunke 08].

No improvement where achieved in context modeling with CART on the Latin nor on Arabic handwriting. This can be explained by the feature extraction which models the context implicitly and the triphone context which may not fit handwriting. The main

changes of the characters regarding their context are made in the height of the connectors to the preceding and succeeding character. In handwriting, the connection of most characters only depends on the preceding character and therefore, the triphone context might not fit.

The results on both databases reported in this work are rated in line with the ones of the best published systems. Thus, the here proposed script independent HWR system has proven to extend the current state of the art in HWR and provides a promising field for further research and replications.

Outlook

The results have shown that the language- and visual model have a great impact and can compensate for most preprocessing steps usually performed. Therefore, the combination of the proposed methods with additional preprocessing and feature extraction may be able to further improve the quality of the visual model. Especially features extracting shape informations such as dots or short strokes may be able to improve the recognition of punctuation- and quotation marks which is still not sufficient.

Another approach to improve the recognition of punctuation- and quotation marks is the usage of better language models. The occurrence of those symbols seems not to be represented well within the ones currently used.

Also, the usage of sub-word recognition has proven to perform well on continuous HWR and was successfully used by [Abdulkadr 06] in the ICDAR 2007 competition. Therefore, subword recognition should be tested on the Arabic single word recognition task.

The triphone context modeling proposed in this work was not able to obtain good results. Nonetheless it is known, that the previous character can influence the shape of a character. Thus, a context modeling using a diphone may be appropriate for handwriting.

Another method to model the variations in the handwritten text is the writer adaptive training (see [Dreuw & Rybach$^+$ 09]). In combination with the writer independent methods proposed in this work further improvements can be expected.

By using recognizer ensembles, [Bertolami & Bunke 08] showed that the recognition results can be improved significantly. This is a very promising approach especially when combining sub-word and word recognizers as, for example, character recognizers are able to recognize any word but not as reliable as a word recognizer. The drawback of the word recognizer is the limited vocabulary. By the combination of both, the reliability of the word recognizer can be combined with the unrestrained recognition of the character recognizer.

At last, the script independence of the proposed system needs to be verified on more scripts and languages as proposed by [Natarajan & Saleem^{+} 08].

List of Figures

List of Tables

Glossary

BCC	Binary Connected Components Analysis
CART	Classification And Regression Tree
CER	Character Error Rate
EM	Expectation Maximization
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HWR	Handwriting Recognition
LDA	Linear Discriminant Analysis
LM	Language Model
MLE	Model Length Estimation
MMI	Maximum Mutual Information
NN	Artificial Neural Network
NN	Nearest Neighbour
OCR	Optical Character Recognition
OOV	Out Of Vocabulary
PCA	Principal Components Analysis
SVM	Support Vector Machine
TDP	Time Distortion Penalty

WER Word Error Rate

Bibliography

[Abdulkadr 06] A. Abdulkadr: Two-tier approach for arabic ofline handwriting recognition. In *International Workshop on Frontiers in Handwriting Recognition*, pp. 161–166, 2006.

[Abed & Märgner 09] H.E. Abed, V. Märgner: Improvement of Arabic handwriting recognition systems; combination and/or reject? *Document Recognition and Retrieval XVI*, Vol. 7247, No. 10, 2009.

[Aburas & Gumah 08] A.A. Aburas, M.E. Gumah: Arabic handwriting recognition: Challenges and solutions. In *International Symposium on Information Technology*, Vol. 2, pp. 1–6, Aug 2008.

[Alma'adeed & Higgens$^+$ 02] S. Alma'adeed, C. Higgens, D. Elliman: Recognition of off-line handwritten Arabic words using hidden Markov model approach. In *International Conference on Pattern Recognition*, Vol. 3, pp. 481–484, 2002.

[Bakis 76] R. Bakis: continuous Speech Word Recognition via Centisecond Acoustic States. In *91st Meeting of the Acoustical Society of America (ASA)*, Washington, DC, USA, 1976.

[Bayes 63] T. Bayes: An Essay towards Solving a Problem in the Doctrine of Chances. *Philosophical Transactions of the Royal Society of Londong*, Vol. 53, pp. 370–418, 1763.

[Bazzi & Schwartz$^+$ 99] I. Bazzi, R. Schwartz, J. Makhoul: An Omnifont Open-Vocabulary OCR System for English and Arabic. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 6, pp. 495–504, 1999.

[Bertolami & Bunke 05] R. Bertolami, H. Bunke: Ensemble methods for handwritten text line recognition systems. In *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 3, pp. 2334–2339 Vol. 3, Oct 2005.

[Bertolami & Bunke 08] R. Bertolami, H. Bunke: Hidden Markov model-based ensemble methods for offline handwritten text line recognition. *Pattern Recognition*, Vol. 41, No. 11, pp. 3452–3460, Nov 2008.

[Beulen & Bransch+ 96] K. Beulen, E. Bransch, M. Kramer, H. Ney: State-Tying für kontextabhängige Phonemmodelle. In *Proc. ITG-Fachtagung Sprachkommunikation*, pp. 51–54, Frankfurt am Main, Germany, Sep 1996.

[Beulen & Bransch+ 97] K. Beulen, E. Bransch, H. Ney: State-Tying for Context Dependent Phoneme Models. In *European Conference on Speech Communication and Technology*, Vol. 3, pp. 1179–1182, Rhodes, Greece, Sep 1997.

[Buck & Gass+ 08] C. Buck, T. Gass, A. Hannig, J. Hosang, S. Jonas, J.T. Peter, P. Steingrube, J.H. Ziegeldorf: Data-Mining-Cup 2007: Vorhersage des Einlöseverhaltens. *Informatik Spektrum*, Vol. 31, No. 6, pp. 591–599, Dec 2008.

[Bunke & Bengio+ 04] H. Bunke, S. Bengio, A. Vinciarelli: Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 6, pp. 709–720, Jun 2004.

[Bunke & Roth+ 95] H. Bunke, M. Roth, E.G. Schukat-Talamazzini: Off-line cursive handwriting recognition using hidden markov models. *Pattern Recognition*, Vol. 28, No. 9, pp. 1399–1413, Sep 1995.

[Caesar & Gloger+ 93] T. Caesar, J. Gloger, E. Mandler: Preprocessing and feature extraction for a handwriting recognitionsystem. In *International Conference on Document Analysis and Recognition*, pp. 408–411, Tsukuba City, Japan, Oct 1993.

[Chellapilla & Simard+ 06] K. Chellapilla, P. Simard, A. Abdulkader: Allograph based writer adaptation for handwritten character recognition. In *International Workshop on Frontiers in Handwriting Recognition*, Oct 2006.

[Dreuw & Heigold+ 09] P. Dreuw, G. Heigold, H. Ney: Confidence-Based Discriminative Training for Model Adaptation in Offline Arabic Handwriting Recognition. In *International Conference on Document Analysis and Recognition*, Barcelona, Spain, Jul 2009.

[Dreuw & Jonas+ 08] P. Dreuw, S. Jonas, H. Ney: White-Space Models for Offline Arabic Handwriting Recognition. In *International Congress on Pattern Recognition*, pp. 1–4, Tampa, Florida, USA, Dec 2008.

[Dreuw & Rybach+ 09] P. Dreuw, D. Rybach, C. Gollan, H. Ney: Writer Adaptive Training and Writing Variant Model Refinement for Offline Arabic Handwriting Recognition. In *International Conference on Document Analysis and Recognition*, Barcelona, Spain, Jul 2009.

[Duda & Hart+ 01] R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*. Wiley-Interscience, 3rd edition, 2001.

[El-Hajj & Likforman-Sulem+ 05] R. El-Hajj, L. Likforman-Sulem, C. Mokbel: Arabic Handwriting Recognition Using Baseline Dependant Features and Hidden Markov Modeling. In *International Conference on Document Analysis and Recognition*, Vol. 1, pp. 70–74, Seoul, Korea, Aug 2005.

[Francis & Kucera 64] W.N. Francis, H. Kucera. *Manual of Information to Accompany a Standard Corpus of Present-Day Edited American English, for use with Digital Computers*. Department of Linguistics, Brown University, Providence, Rhode Island, USA, 1964. http://icame.uib.no/brown/bcm.html.

[Gupta & Niranjan+ 06] G. Gupta, S. Niranjan, A. Shrivastava, R. Sinha: Document Layout Analysis and Classification and Its Application in OCR. In *Enterprise Distributed Object Computing Conference Workshops, 2006. EDOCW '06. 10th IEEE International*, pp. 58–58, Oct 2006.

[Heigold & Deselaers+ 08] G. Heigold, T. Deselaers, R. Schlüter, H. Ney: Modified MMI/MPE: A Direct Evaluation of the Margin in Speech Recognition. In *International Conference on Machine Learning*, pp. 384–391, Helsinki, Finland, Jul 2008.

[Holmes & Vine+ 98] J. Holmes, B. Vine, G. Johnson. *Guide to the Wellington Corpus of Spoken New Zealand English*. School of Linguistics and Applied Language Studies, Victoria University of Wellington, Wellington, New Zealand, Jun 1998.

[Johansson & Leech+ 78] S. Johansson, G.N. Leech, H. Goodluck. *Manual of Information to Accompany the Lancaster-Oslo/Bergen Corpus of British English, for Use With Digital Computers*. Department of English, University of Oslo, Oslo, Norway, Dec 1978. http://khnt.hit.uib.no/icame/manuals/lob/INDEX.HTM.

[Juan & Toselli+ 01] A. Juan, A.H. Toselli, J. Domnech, J. González, I. Salvador, E. Vidal, F. Casacuberta: Integrated Handwriting Recognition and Interpretation via Finite-State Models. *International Journal of Pattern Recognition and Artifial Intelligence*, Vol. 2004, pp. 519–539, 2001.

[Keysers 07] D. Keysers: Comparison and Combination of State-of-the-art Techniques for Handwritten Character Recognition: Topping the MNIST Benchmark. techreport, IUPR Research Group, DKFI and TU Kaiserslautern, Oct 2007.

[Keysers & Dahmen+ 00] D. Keysers, J. Dahmen, H. Ney: A Probabilistic View on Tangent Distance. In *Deutsche Arbeitsgemeinschaft für Mustererkennung Symposium*, pp. 107–114, Kiel, Germany, Sep 2000. Springer.

[Keysers & Deselaers+ 07] D. Keysers, T. Deselaers, C. Gollan, H. Ney: Deformation Models for Image Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 8, pp. 1422–1435, Aug 2007.

[Kneser & Ney 95] R. Kneser, H. Ney: Improved Backing-off for m-gram Language Modeling. In *International conference on Acoustics, Speech and Signal Processing*, Vol. 1, pp. 181–184, 1995.

[Koerich & Sabourin$^+$ 03] A.L. Koerich, R. Sabourin, C.Y. Suen: Large Vocabulary Off-Line Handwriting Recognition: A Survey. *Pattern Analysis and Applications*, Vol. 6, pp. 97–121, 2003.

[Lee 96] S.W. Lee: Off-Line Recognition of Totally Unconstrained Handwritten Numerals Using MCNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, pp. 648–652, 1996.

[Liang 83] M. Liang: *Word Hy-phen-a-tion by Com-put-er*. Ph.D. thesis, Department of Computer Science, Stanford University, Stanford, CA, USA, Aug 1983.

[Lorigo & Govindaraju 06] L. Lorigo, V. Govindaraju: Offline Arabic handwriting recognition: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 5, pp. 712–724, May 2006.

[Lööf & Gollan$^+$ 07] J. Lööf, C. Gollan, S. Hahn, G. Heigold, B. Hoffmeister, C. Plahl, D. Rybach, R. Schlüter, H. Ney: The RWTH 2007 TC-STAR Evaluation System for European English and Spanish. In *Interspeech 2007*, pp. 2145–2148, Antwerp, Belgium, Aug 2007.

[Majidi 06] M.R. Majidi: *Einführung in die arabisch-persische Schrift*. Helmut Buske Verlag, 2006.

[Mandler & Oberländer 89] E. Mandler, M. Oberländer: A single pass algorithm for fast contour coding of binary images. In *DAGM-Symposium*, 1989.

[Märgner & Abed 07] V. Märgner, H.E. Abed: Arabic Handwriting Recognition Competition. In *International Conference on Document Analysis and Recognition*, Vol. 2, pp. 1274–1278, Sep 2007.

[Märgner & Pechwitz$^+$ 05] V. Märgner, M. Pechwitz, H. Abed: Arabic handwriting recognition competition. In *International Conference on Document Analysis and Recognition*, Vol. 1, pp. 70–74, Seoul, Korea, Aug 2005.

[Marti & Bunke 99] U.V. Marti, H. Bunke: A full English sentence database for off-line handwriting recognition. In *International Conference on Document Analysis and Recognition*, pp. 705 – 708, 1999.

[Marti & Bunke 02a] U.V. Marti, H. Bunke: The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, Vol. 5, No. 1, pp. 39–46, Nov 2002.

[Marti & Bunke 02b] U.V. Marti, H. Bunke: Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition systems. *World Scientific Series In Machine Perception And Artificial Intelligence*, Vol. 1, pp. 65–90, 2002.

[Menasri 08] F. Menasri: Reconnaissance de l'écriture arabe manuscrite. In *Séminaire DGA/DET*, Jun 2008.

[Mozaffari & Faez$^+$ 07] S. Mozaffari, K. Faez, V. Margner, H. El Abed: Strategies for Large Handwritten Farsi/Arabic Lexicon Reduction. In *International Conference on Document Analysis and Recognition*, Vol. 1, pp. 98–102, Sep 2007.

[Natarajan & Saleem$^+$ 08] P. Natarajan, S. Saleem, R. Prasad, E. MacRostie, K. Subramanian: *Arabic and Chinese Handwriting Recognition*, Vol. 4768/2008 of *LNCS*, chapter Multi-lingual Offline Handwriting Recognition Using Hidden Markov Models: A Script-Independent Approach, pp. 231–250. Springer Berlin / Heidelberg, 2008.

[Neubauer 96] M. Neubauer: Feinheiten bei wissenschaftlichen Publikationen - Mikrotypographie Regeln, Teil 1. *Die TEXnische Kommödie*, Vol. 4, pp. 23–40, 1996.

[Ney 90] H. Ney: Acoustic Modeling of Phoneme Units for Continuous Speech Recognition. In *5th European Signal Processing Conference, Signal Processing V: Theories and Applications*, pp. 65–72. Elsevier Science Publishers, Dec 1990.

[Nopsuwanchai & Povey 03] R. Nopsuwanchai, D. Povey: Discriminative training for HMM-based offline handwritten character recognition. In *International Conference on Document Analysis and Recognition*, pp. 114–118, 2003.

[Pechwitz & Maddouri$^+$ 02] M. Pechwitz, S.S. Maddouri, V. Märgner, N. Ellouze, H. Amiri: IFN/ENIT-Database of Handwritten Arabic Words. In *Colloque International Francophone sur l'Ecrit et le Document*, pp. 129–136, Hammamet, Tunis, Oct 2002.

[Pechwitz & Märgner 03] M. Pechwitz, V. Märgner: HMM Based Approach for Handwritten Arabic Word Recognition Using the IFN/ENIT - Database. In *International Conference on Document Analysis and Recognition*, pp. 890– 894, Edinburgh, Scottland, Aug. 2003.

[Pitrelly & Roy 03] J.F. Pitrelly, A. Roy: Creating word-level language models for large-vocabulary handwriting recognition. *International Journal on Document Analysis and Recognition*, Vol. 5, pp. 126–137, 2003.

[Plamondon & Srihari 00] R. Plamondon, S. Srihari: Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 1, pp. 63–84, Jan 2000.

[Romero & Alabau$^+$ 07] V. Romero, V. Alabau, J.M. Benedi: Combination of N-Grams and Stochastic Context-Free Grammars in an Offline Handwritten Recognition System. *Lecture Notes in Computer Science*, Vol. 4477, pp. 467–474, 2007.

[Rybach 06] D. Rybach: Appearance-Based Features for Automatic Continuous Sign Language Recognition. Master's thesis, Human Language Technology and Pattern Recognition Group, RWTH Aachen University, Aachen, Germany, Jun 2006.

[Rybach & Gollan$^+$ 09] D. Rybach, C. Gollan, G. Heigold, B. Hoffmeister, J. Lööf, R. Schlüter, H. Ney: The RWTH Aachen University Open Source Speech Recognition System. In *Interspeech*, Brighton, U.K., Sep 2009.

[Schambach 03] M.P. Schambach: Model length adaptation of an HMM based cursive word recognition system. In *International Conference on Document Analysis and Recognition*, Vol. 1, pp. 109– 113, Edinurgh, Scottland, Aug 2003.

[Schambach & Rottland$^+$ 08] M.P. Schambach, J. Rottland, T. Alary: How to Convert a Latin Handwriting Recognition System to Arabic. In *Internation Conference on Frontiers in Handwriting Recognition*, 2008.

[Shafait & van Beusekom$^+$ 08] F. Shafait, J. van Beusekom, D. Keysers, T. Breuel: Background variability modeling for statistical layout analysis. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pp. 1–4, Dec 2008.

[Shepard 53] D.H. Shepard: Apparatus for reading, Dec 1953. US-Patent No 2663758.

[Srihari 93] S. Srihari: Recognition of HAndwritten and Machineprinted Text for Postal Adress Interpretation. *Pattern Recognition Letters*, Vol. 14, pp. 291–302, 1993.

[Stolcke 02] A. Stolcke: SRILM - An Extensible Language Modeling Toolkit. In *International Conference on Spoken Language Processing*, Vol. 2, pp. 901–904, Denver, CO, USA, Sep 2002.

[Sun & Si 97] C. Sun, D. Si: Skew and slant correction for document images using gradient direction. In *International Conference on Document Analysis and Recognition*, Vol. 1, pp. 142–146, Aug 1997.

[Tappert & Suen$^+$ 90] C. Tappert, C. Suen, T. Wakahara: The state of the art in online handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 12, No. 8, pp. 787–808, Aug 1990.

[Verma & Blumenstein$^+$ 98] B. Verma, M. Blumenstein, S. Kulkarni: Recent Achievements In Off-Line Handwriting Recognition Systems. In *In Proceedings of the International Conference on Computational Intelligence and Multimedia Applications*, pp. 27–33, 1998.

[Vinciarelli & Luettin 01] A. Vinciarelli, J. Luettin: A new normalization technique for cursive handwritten words. *Pattern Recognition Letters*, Vol. 22, No. 9, pp. 1043–1050, Jul 2001.

[Yanikoglu & Sandon 98] B. Yanikoglu, P.A. Sandon: Segmentation of off-line cursive handwriting using linear programming. In *Pattern Recognition*, Vol. 31, pp. 1825–1833, 1998.

[Zhang & Jin$^+$ 03] J. Zhang, R. Jin, Y. Yang, A. Hauptmann: Modified logistic regression: An approximation to XVM and its applications in large-scale text categorization. In *International Conference on Machine Learning*, Aug 2003.

[Zimmermann & Bunke 02] M. Zimmermann, H. Bunke: Hidden Markov model length optimization for handwriting recognition systems. In *International Workshop on Frontiers in Handwriting Recognition*, pp. 369–374, 2002.