

# **Ensemble Methods for Offline Handwritten Text Line Recognition**

Inauguraldissertation  
der Philosophisch-naturwissenschaftlichen Fakultät  
der Universität Bern

vorgelegt von

**Roman Bertolami**

von Thun, BE

Leiter der Arbeit:

Prof. Dr. H. Bunke

Institut für Informatik und angewandte Mathematik  
Universität Bern



# **Ensemble Methods for Offline Handwritten Text Line Recognition**

Inauguraldissertation  
der Philosophisch-naturwissenschaftlichen Fakultät  
der Universität Bern

vorgelegt von

**Roman Bertolami**

von Thun, BE

Leiter der Arbeit:

Prof. Dr. H. Bunke

Institut für Informatik und angewandte Mathematik  
Universität Bern

Von der Philosophisch-naturwissenschaftlichen Fakultät angenommen.

Bern, den 22. Mai 2008

Der Dekan:

Prof. Dr. Paul Messerli



# Abstract

This thesis investigates ensemble methods for offline recognition of English handwritten text lines. Multiple recognisers are automatically generated from a single base recognition system. Combining the output of these multiple recognisers provides the final ensemble result.

The underlying recognisers are based on hidden Markov models. One model is built for each character. Based on the lexicon, word models are derived by concatenating character models. A statistical language model is used to build text line models by preferring more likely word sequences over unlikely word sequences. A postprocessing step calculates confidence values for each recognised word.

Ensembles of recognisers are generated based on variation of the training data, the features, and the system architecture. Because the output of a handwritten text line recogniser is a sequence of words, most existing combination methods cannot be applied directly. The combination has to be performed in two steps. First, the word sequences are synchronised by a string alignment procedure. Second, a decision strategy derives the combination result for each segment of the alignment. For this purpose, confidence-based voting, a statistical decision method, and a decision method that includes language model information are used.

The experimental evaluation on a large set of images of handwritten text lines indicates that the proposed ensemble methods can significantly increase the performance of an offline handwritten text line recognition system.



# Acknowledgments

First, I thank Prof. Dr. Horst Bunke as the supervisor of this thesis. I also thank Prof. Dr. Guy Lorette for the willingness to act as co-referee of this thesis and Prof. Dr. Hanspeter Bieri for supervising the defence examination.

Many thanks to Matthias Zimmermann for introducing me to the field of handwriting recognition and for numerous interesting discussions on this subject. I am grateful for the interesting collaboration on slant correction with Prof. Dr. Seiichi Uchida from the Kyushu University in Japan. For the proof-reading of this thesis, I thank Shiva Grings, Michel Neuhaus, and Anita Bertolami.

I thank my colleagues from the University of Bern who helped with advice, discussions, support, and coffee breaks. Peppo Brambilla, Thomas Buchberger, Andreas Fischer, Alicia Fornés, Volkmar Frinken, Christoph Gutmann, Emanuel Indermühle, Christophe Irniger, Vivian Kilchherr, Marcus Liwicki, Michel Neuhaus, Kaspar Riesen, Philippe Robert, Sonja Schär, Andreas Schlapbach, Barbara Spillmann, Susanne Thüler, Tamás Varga, and Thomas Wenger. Thank you very much.

Special thanks to my mother Beatrice, to my father Dario, to my sisters Anita and Livia, and to Barbara. You have been a great support for me during this time.

Many thanks to my numerous friends not mentioned above who made my life pleasant even in some of the hardest times, especially to Alain, Andrea, Beppe, Christoph, Ehsan, Eveline, Filiz, Gregor, Isa, Jonas, Marc, Markus, Mathias, Mick, Mike, Monika, Norbert, Stefan, Peter, Sibylle, Simon, Susanne, Tobias, and Ute.

Last but not least, this research was supported by the Swiss National Science Foundation (Nr. 20-52087.97). Additional funding was provided by the Swiss National Science Foundation NCCR program “Interactive Multimodal Information Management (IM)<sup>2</sup>”. Further support was given by “Stiftung zur Förderung der wissenschaftlichen Forschung an der Universität Bern”. I thank them for their financial support.





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Problem Statement and Goal . . . . .	2
1.3	Contribution . . . . .	2
1.4	Outline of the Thesis . . . . .	3
<b>Part I — Handwritten Text Line Recognition</b>		
<b>2</b>	<b>Introduction</b>	<b>7</b>
2.1	Challenges . . . . .	7
2.2	State of the Art . . . . .	10
2.3	The IAM Database . . . . .	12
2.4	System Overview . . . . .	13
<b>3</b>	<b>Handwriting Recognition Based on Hidden Markov Models</b>	<b>17</b>
3.1	Image Normalisation . . . . .	17
3.2	Feature Extraction . . . . .	22
3.3	Hidden Markov Modelling . . . . .	23
3.4	Evaluation Methodology . . . . .	27
<b>4</b>	<b>Language Modelling</b>	<b>31</b>
4.1	English Text Corpora . . . . .	31
4.2	Fundamentals of $N$ -Gram Language Models . . . . .	32
4.3	Smoothing . . . . .	33
4.4	Language Models and Hidden Markov Model Decoding . . . . .	35
<b>5</b>	<b>Confidence Measures</b>	<b>39</b>
5.1	Likelihood-Based Confidence . . . . .	40
5.2	Confidence Derived from Candidates . . . . .	40
5.3	Rejection Strategy . . . . .	43
<b>6</b>	<b>Experimental Evaluation</b>	<b>45</b>
6.1	Experimental Setup . . . . .	45
6.2	Reference Recogniser . . . . .	46

6.3	Non-Uniform Slant Correction . . . . .	50
6.4	Confidence-Based Rejection . . . . .	52
6.5	Discussion . . . . .	53

## Part II — Ensemble Methods

<b>7</b>	<b>Introduction</b>	<b>59</b>
7.1	Challenges . . . . .	60
7.2	State of the Art . . . . .	61
7.3	System Overview . . . . .	64
<b>8</b>	<b>Ensemble Generation</b>	<b>67</b>
8.1	Bagging . . . . .	67
8.2	Feature Subspace . . . . .	68
8.3	Language Model Variation . . . . .	69
8.4	Ensemble Member Selection . . . . .	70
<b>9</b>	<b>Result Combination</b>	<b>73</b>
9.1	Receiver Output Voting Error Reduction . . . . .	74
9.2	Statistical Decision . . . . .	76
9.3	Language Model Decision . . . . .	77
9.4	Oracle . . . . .	82
<b>10</b>	<b>Diversity Analysis</b>	<b>83</b>
10.1	Diversity Measures . . . . .	84
10.2	Diversity Analysis for Word Sequence Recognisers . . . . .	85
<b>11</b>	<b>Experimental Evaluation</b>	<b>89</b>
11.1	Experimental Setup . . . . .	89
11.2	ROVER Combination . . . . .	90
11.3	Statistical Decision . . . . .	92
11.4	Language Model Decision . . . . .	94
11.5	Oracle . . . . .	96
11.6	Diversity Analysis Results . . . . .	97
11.7	Discussion . . . . .	104

## Part III — Conclusions and Outlook

<b>12</b>	<b>Conclusions</b>	<b>109</b>
<b>13</b>	<b>Outlook</b>	<b>111</b>
	<b>Bibliography</b>	<b>113</b>

# 1

---

## Introduction

### 1.1 Motivation

Cursive handwriting recognition has been an active research topic for many years. The recognition of isolated characters or digits is quite mature and has been integrated into many applications. However, if the complexity of the recognition task increases and entire words are considered instead of single characters, recognition rates drop significantly because the number of characters in a word is unknown in advance and the considered lexicon is typically large. Even more difficult is unconstrained handwritten text line recognition, especially if a writer independent task is considered, and no training data from the person who has written the input is available. The main problems in this field are the differences in writing style, the large lexicon, and the segmentation problem, i.e. the unknown number and position of the words on a text line.

The reading of human handwriting by machines is an interesting and intellectually challenging problem. The aim to approach the performance of humans in handwritten text recognition is a major driving force behind many research activities in this field. Additionally, handwriting recognition has become important from the application oriented point of view. Most industrial applications using offline handwriting recognition are only able to solve constrained and specific problems, e.g. address reading and bank cheque processing. Moreover, the automatic reading of general text is interesting for tasks such as the transcription of handwritten historical archives and the automatic reading of forms, handwritten faxes, personal notes, and annotations on documents.

Recognition accuracy in pattern recognition can often be improved if multiple classifiers methods are used. Of particular interest are ensemble methods, where different classifiers are automatically generated from a base classifier by systematically modifying either the training data, the features, or the system architecture. In a multiple classifier system, each of the individual systems classifies the input pattern first. Next,

a combination module generates the final result based on the results of the individual classifiers. Such a multiple classifier approach often achieves a better recognition performance than a single classifier.

However, most of the combination techniques applied in current multiple classifier systems are not suitable for the combination of handwritten text line recognisers because the output of a text line recogniser is a sequence of words rather than just a single class, and the number of words in the output word sequence may differ in the multiple recognition results.

## **1.2 Problem Statement and Goal**

In this thesis the problem of writer independent unconstrained offline handwritten text line recognition is addressed. Until today only a few works have been published in this area, and none of them includes a systematic investigation of applying ensemble methods in order to improve recognition performance.

The goal of this thesis is to explore the capability of ensemble methods to improve a state of the art system to recognise unconstrained offline handwritten English text lines.

## **1.3 Contribution**

The main contribution of this thesis is the investigation of ensemble methods for unconstrained offline handwritten text line recognition including extensive experimental evaluation of the developed methods. The following subsections provide a more detailed list of the contributions.

### **1.3.1 Handwritten Text Line Recogniser**

Text line recognisers developed in the past are usually based on the closed lexicon assumption, i.e. each word that occurs in the test set is present in the lexicon. This assumption is now dropped. The novel approach determines the lexicon based on large text corpora and not based on the test set. This is a more realistic scenario because the words to be recognised are usually unknown in advance.

Novel confidence measures based on alternative candidates derived from a specific integration of a language model are calculated and applied in a combination context.

A novel non-uniform slant correction technique is applied for the first time to handwritten text lines. By experimental evaluation, the impact of this novel slant correction technique on the recognition performance is examined.

### 1.3.2 Ensemble Generation

Two well-known ensemble generation methods, namely Bagging and the feature subspace method, are applied for the first time to handwritten text line recognition.

A novel ensemble generation method is developed that is specific for handwritten text line recognition. Multiple recognisers are built by varying the level of integration of the statistical language model into the recognition process.

Existing diversity measures cannot be applied directly to ensembles of handwritten text line recognisers. A new generic framework is developed that enables the application of existing diversity measures to these ensembles.

### 1.3.3 Result Combination

A combination framework called ROVER, which was developed in the field of continuous speech recognition, is applied for the first time to combine multiple handwritten text line recognisers.

The ROVER framework is extended by two new decision methods, namely a statistical decision method and decision algorithm which uses language model information. The statistical decision addresses the problem that the outputs of different recognisers may be dependent, whereas the language model decision considers not only the words at the current position, but also the word at previous positions.

## 1.4 Outline of the Thesis

The thesis consists of two main parts representing handwriting recognition and ensemble methods, and a concluding part providing a summary and outlook.

### Part I – Handwritten Text Line Recognition

The first part of this thesis is devoted to offline handwritten text line recognition. After an introduction to the field, a recognition system is described which serves as base recogniser in the second part of the thesis. The recognition system can be divided into three parts: preprocessing, including image normalisation and feature extraction, recognition, which is based on hidden Markov models and statistical language modelling, and postprocessing, where confidence measures are calculated.

## **Part II – Ensemble Methods**

Ensemble methods are applied to handwritten text line recognition in the second part. After an introduction to ensemble methods with a special focus on applications in handwriting recognition, several methods to generate multiple handwritten text line recognisers are presented. The word sequences that are the output of the multiple recognisers are finally combined by various combination methods.

## **Part III – Conclusions and Outlook**

Conclusions are drawn and an outlook on future research is provided in the last part of this thesis.

**Part I**

**Handwritten Text Line  
Recognition**





# 2

---

## Introduction

This part presents a recognition system for unconstrained offline handwritten text line recognition. This system acts as base recogniser in the ensemble methods that will be introduced in Part II. The recognition system is based on the recognition systems presented in [89, 153]. Several enhancements have been introduced compared to [89] including individual length optimisation of the character models and optimisation of the integration of the statistical language model. In contrast to [153], text lines are considered instead of sentences, which is a more general approach because less constraints are imposed to the input data. Additionally, the lexicon is not closed over the test set anymore which represents a more realistic scenario because the words to be recognised are usually unknown in advance.

The next section of this chapter discusses the main challenges that must be addressed in the field of unconstrained offline handwritten text line recognition. The state of the art including character, word, and text recognition is summarised in Sect. 2.2. Section 2.3 presents the handwriting image data used throughout this thesis, before a system overview is given in the last section of this chapter.

The remaining chapters of this part are organised as follows. Chapter 3 describes the recognition system including image normalisation, feature extraction, and hidden Markov modelling methodology. Next, natural language modelling techniques are presented in Chapter 4. Confidence measure methodology is covered by Chapter 5. The last chapter of this part reports the experimental evaluation of the presented techniques.

### 2.1 Challenges

One goal of the presented recognition system is to be as general as possible, i.e. to impose only few restriction to the handwriting to be recognised. This is contrary to

task specific systems, e.g. bank cheque readers, where many assumptions can be made about the input data, which often simplifies the recognition.

The following subsections present the main challenges of the considered recognition task and describe how the proposed handwritten text line recogniser addresses these challenges.

### **2.1.1 Offline Recognition**

In handwriting recognition one distinguishes between offline and online recognition. The difference originates from the type of input data that is available for recognition. In online recognition a special input device, e.g. an electronic pen, tracks the movement of the pen during the writing process. Based on this information, an online recognition system can determine the position of a pen at a given time and hence order the input points in a temporal sequence describing how the writing occurred.

No time information is available in offline handwriting recognition. Only an image of the handwriting is processed. Thus, the writing process cannot be completely reconstructed. Furthermore, no constraint is given in terms of the writing instrument (e.g. quill) and writing pad (e.g. paper), as long as the scanned image shows enough contrast to distinguish foreground from background pixels. Typically, a handwriting document is scanned as a greyscale image at a resolution of about 300 dpi, which results in a relatively low-quality image of handwritten text.

Because less information is available, offline recognition is usually considered more difficult than online recognition. Furthermore, offline recognition is more human-like because only the image information is available when humans read handwritten texts. In this thesis only offline handwriting recognition is addressed. However, many methods apply directly to online handwriting recognition, too.

### **2.1.2 Unconstrained Text Recognition**

To act as a general recogniser, the proposed system requires as few conditions on the handwritten input data as possible. The only constraint is that the texts must be written in English because the lexicon is obtained from English text corpora.

In contrast, recognisers built for specific application domains can use task specific knowledge to improve recognition accuracy; e.g. in bank cheque processing the legal and the courtesy amount of the cheque must be equivalent. Another example are address reading systems which have a specific lexicon containing only postal codes, city names, and street names. Additionally, relations between postal codes and city names, as well as between street names and cities can be used to improve performance.

To deal with unconstrained texts the lexicon of a recogniser must be large enough, representing a high coverage of the words that may occur in a text. On the other hand,

if the lexicon is too large, the recognition may become computationally expensive and unreliable because many similar words must be distinguished. Some works overcome this problem by closing the lexicon over the test set, i.e. it is assured that each word that occurs in the test set is present in the lexicon. In general, however, the words in the test set are unknown, and the scenario with a lexicon that is not closed over the test set is more realistic.

### 2.1.3 Segmentation Problem

Because the correct number of words in a text line is unknown in advance, the recogniser is often unable to detect the beginning and the ending of a word correctly. This leads to word segmentation errors because too many or too few words are present in the recognised word sequence.

A similar segmentation problem occurs when words are split into characters, which is known as Sayre's paradox [116]. A letter cannot be recognised before being correctly segmented and cannot be segmented before being recognised.

To overcome the paradox, the recogniser used in this thesis is based on hidden Markov models and implicitly segments the text line into words and characters when performing the recognition based on probabilistic models.

### 2.1.4 Writer Independent Recognition

Writer independent recognition implies that no handwriting sample of a writer in the test set is available to train and validate the recognition system. Writer independence aims at the highest possible generalisation regarding writing styles and instruments. In applications such as address reading or bank cheque processing, the handwriting must be recognised independently of the writer, because it is not possible to obtain handwritten samples from each customer.

The opposite of writer independent recognition is single-writer or multi-writer recognition. Such a recognition system is optimised for a given set of writers and will usually perform very badly on handwritten samples from other people. Writer dependent methods are typically applied if only a few people are using a recognition system, e.g. in a personalised mobile device.

In this thesis only writer independent recognition is addressed. Independent recognition is more general and considered to be more challenging because of the different writing styles and instruments. Figure 2.1 shows some examples of different writing styles illustrating the difficulties a writer independent recognition system must address.

Charles obliged with "April Serenade". This week it appears,  
 The Government should settle this argument with two words to  
 on new techniques - and on the universities to  
 Vaughan will burst on to the London Palladium stage  
 think hard and long before his next jump into the  
 I don't think he will storm the charts with this one, but it's a good start.  
 He double-crosses the five pals with whom he lives,

**Figure 2.1** *Examples of different writing styles.*

## 2.2 State of the Art

This section summarises state of the art cursive Roman handwriting recognition systems. The focus is on offline handwriting systems for isolated characters, for cursive words, and for general text. A brief historical overview of handwriting recognition is provided in [83]. Extensive surveys of handwriting recognition research are given in [15, 102].

### 2.2.1 Isolated Character Recognition

The problem of isolated character recognition is to assign each input image to a character class. It is usually treated as a traditional pattern classification problem. Typical classes are all upper and lower case characters as well as the ten digits. A survey of earlier work in the domain of isolated character recognition is provided in [57]. A more recent overview is given in [22].

Several databases, including NIST [38], ETL-6 [114], and CCC [17], became publicly available and are widely used to develop various isolated character recognition systems. Almost all generic classifier methods have been applied to isolated character recognition. Examples include  $k$ -nearest-neighbour classifiers [122], neural networks [74], and support vector machines [79]. In the last two years, among other

works, [17, 18, 21, 25, 53, 110] have addressed the problem of isolated character recognition.

It is worth noting that isolated Roman character recognition is quite mature today. Currently much research is focussed on isolated characters of non-Roman alphabets like Arabic/Farsi [84], Chinese [123], Hangul [61], and others.

### 2.2.2 Word Recognition

The recognition of entire words is considered a more difficult task than isolated character recognition because, usually, a word must be segmented into characters first. However, the segmentation of a word into characters is difficult if no knowledge about the word is available (a problem known as Sayre's paradox [116]).

Several databases are available to develop and test handwritten word recognisers. The IRONOFF database [133] contains online and offline signals of handwritten words for small-lexicon tasks. Many word recognition systems are developed using postal address data from the CEDAR database [55]. The segmented version of the IAM database [90] provides English words extracted from handwritten texts.

Surveys of previous work in the field of offline handwritten word recognition can be found in [15, 72, 125, 134]. Three main approaches have been used to address the word recognition problem: the holistic approach, the segmentation-based approach, and the segmentation-free approach. These will be described in more details in the following paragraphs.

Holistic methods overcome the difficult segmentation problem by classifying the given input image directly as a word of the lexicon [85]. Features that characterise the word as a whole are extracted from the image and compared to known prototypes. Limited to a small number of word classes, holistic methods cannot handle large-lexicon problems.

A segmentation-based approach first segments a word into smaller entities, e.g. characters or graphemes. These entities are then classified with a character recognition system and recombined to build the recognised word. This recombination is usually based on heuristic rules driven by human intuition. Examples of such procedures include [7, 64, 67, 80].

Most recent work on handwritten word recognition implements segmentation-free methods based on hidden Markov models. The segmentation and the recognition step are performed concurrently in a single step. Models are built for each character, and these models are then concatenated to word models. Among others, recent work using a segmentation-free approach to recognise offline handwritten words includes [1, 39, 40, 47, 66, 71].

### 2.2.3 Text Recognition

Unconstrained handwritten text recognition is still considered a very challenging task with many open problems. Only relatively few published work address unconstrained handwritten text recognition.

A segmentation-based approach has been proposed in [63] to address the task of unconstrained handwritten text recognition. After image normalisation the text lines are segmented into words and characters, respectively. The segmented characters are then classified using a handwritten character recogniser. A tree-based segmentation method has been used in [130] to separate a text line into words. A word recogniser based on hidden Markov models then performs the recognition.

Several segmentation-free approaches have been proposed. In [136], a text line recognition system based on hidden Markov models and statistical language models is described. A sliding window approach extracts 16 pixel-based features at each position of the window, which moves from left to right over the image. The lexicon size is systematically varied between 10,000 and 50,000 word classes. In [32], features based on principle component analysis and discrete wavelet transforms are extracted and used by a hidden Markov model based recogniser. Not text lines, but sentences are recognised in [153], where recognition performance is improved by integrating a stochastic context-free grammar into the recognition process.

In [128], an interactive system to transcribe handwritten sentences has been described. A human transcriber corrects wrongly recognised words online and hence helps the recogniser to improve the recognition performance on the remaining input.

Related work has been conducted in the field of online handwriting recognition, too. In [103], a segmentation-based recogniser using neural networks recognises handwritten sentences. The recogniser outputs multiple hypotheses which are rescored with a statistical language model in a postprocessing step. Text lines written on a whiteboard are processed in [82]. A special pen is used to track the movement of the handwriting on the whiteboard. Hidden Markov models are then used to perform the recognition.

## 2.3 The IAM Database

Having a large database of handwritten text lines available is extremely important for the development and evaluation of handwriting recognition systems. Most public offline handwriting databases are restricted to character or isolated word images for a specific domain like bank cheque processing (CENPARMI database [126]) or address reading (CEDAR database [55]). Only few databases are available containing cursive handwritten text. A single-writer database containing about 4,000 words has been used in [118] for conducting writer dependent experiments. The NIST database [38]

is limited in terms of the covered text. Only the first few sentences of the American constitution are provided, which represents only a very small lexicon.

The IAM database [90] covers a large amount of unconstrained English texts. Since its first version it has been extended and enhanced to support different tasks, including isolated word recognition, and writer identification [149]. Currently, the database consists of more than 1,500 forms of handwritten text written by over 600 different writers. These forms contain over 13,000 text lines and more than 12,000 different word instances. The sentences written by the contributors originate from the LOB corpus [60].

Figure 2.2 shows an example of a completed form of the IAM database. The sentences from the LOB corpus were printed on forms, and subjects were asked to write the sentences on the empty area below. No constraints were given concerning writing style and instrument.

## 2.4 System Overview

Figure 2.3 provides an illustrative overview of the recognition system used in this thesis. The system can be divided into three phases: preprocessing, recognition, and postprocessing. Each of them will be described briefly in the following paragraphs and in more detail in the remaining chapters of this part of the thesis.

The goal of the preprocessing phase is to convert a handwritten input image into a feature vector sequence. First, the image is normalised to reduce the impact of different writing styles. Normalisation includes skew correction, slant correction, baseline normalisation, and average character width normalisation. The normalised image is then converted into a feature vector sequence with a sliding window approach. The window moves from left to right over the normalised image of handwritten text. At each position of the window nine geometrical features are extracted.

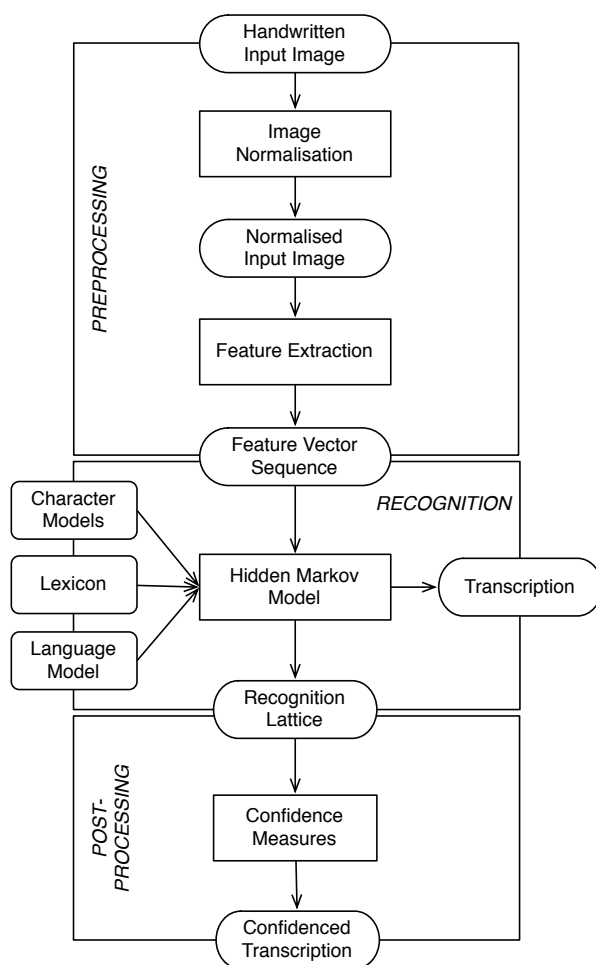
In the recognition phase, the feature vector sequence is decoded using hidden Markov models. One model is built for each character. Based on a lexicon, word models are constructed by concatenating character models. Text line models are then created using these word models and a statistical language model. Using the text line models, the Viterbi algorithm performs the decoding of the feature vector sequence providing either a transcription containing the most likely word sequence or a recognition lattice that consists of an entire network of promising results.

The recognition lattice is used to calculate confidence measures for each recognised word in the postprocessing phase. The confidence measures are derived from alternative candidates that are extracted from the recognition lattice and can either be used for rejection or to support the combination of multiple recognisers.

Sentence Database	G06-011
<p>By the end of the month he still delighted in Naples. He told Cloncurry that he enjoyed it as much as his health permitted him to enjoy anything. 'The Pearl', he wrote, 'is arrived, which is a great resource. Vesuvius seems to be tired; he is going out fast.... What a gay, lively people, and what a busy town. At Rome, every other man was a priest: here the priest is superseded by the soldier - a favourable change in my eye, particularly as the troops are very fine.'</p>	
<p>By the end of the month he still delighted in Naples. He told Cloncurry that he enjoyed it as much as his health permitted him to enjoy anything. 'The Pearl', he wrote, 'is arrived, which is a great resource. Vesuvius seems to be tired; he is going out fast.... What a gay, lively people, and what a busy town. At Rome, every other man was a priest: here the priest is superseded by the soldier - a favourable change in my eye, particularly as the troops are very fine.'</p>	
<p>Name: ROMAN BORTOLANO</p>	

Figure 2.2 Completed form from the IAM database.





**Figure 2.3** System overview including preprocessing, recognition, and postprocessing.



# 3

---

## Handwriting Recognition Based on Hidden Markov Models

This chapter describes the main parts of the handwriting recognition system. The input data of the recognition system are text line images, thus no text line extraction method is described. The sections are organised as follows. After the image normalisation, which includes several procedures to reduce the impact of individual writing styles, the feature extraction process is described in Sect. 3.2. Next, Sect. 3.3 describes the theory of hidden Markov models and its application to handwriting recognition. The last section of this chapter provides the evaluation methodology used to measure the performance of a recogniser.

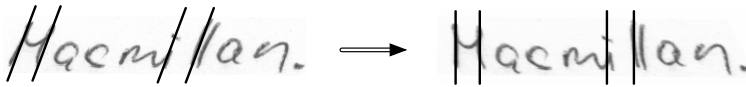
### 3.1 Image Normalisation

To reduce the variability of different writing styles, the images of handwritten text lines are normalised. Several independent procedures are applied to the handwritten images including skew correction, slant correction, writing regions normalisation, and average character width normalisation.

The normalisation is conducted uniformly with global correction parameters for the entire text line. Local deviations are not normalised with these techniques; e.g. the determined slant angle is an average slant angle of the entire text line. However, different letters in the text line might have different slants which are not removed by a global slant correction technique. Therefore, a non-uniform slant correction technique is proposed in Sect. 3.1.5 to enable the correction of local slant angles.



**Figure 3.1** Example illustrating the skew correction. The image on the left side is the original input image. The imaginary line on which the handwriting is written is estimated. A rotation corrects the line to become horizontal. The result is shown in the right image.



**Figure 3.2** Example of the slant correction. The average slant angle is estimated in the skew-corrected image on the left. A shear transformation corrects the estimated slant. The corrected image with upright handwriting is shown on the right.

### 3.1.1 Skew Correction

The skew correction procedure rotates the text line such that the imaginary line on which the words are written becomes horizontal. For this purpose two steps are applied. First, the skew angle is estimated. Second, the input image is rotated by the estimated skew angle. An example that illustrates the skew correction is provided in Fig. 3.1. The motivation of the skew correction is to simplify the estimation of the baselines for the normalisation of the writing regions described below.

### 3.1.2 Slant Correction

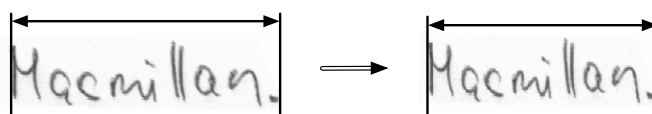
The goal of slant correction is to bring the handwriting into an upright position. Depending on the writing style, the handwriting, especially its long vertical strokes, is more or less slanted to the right or to the left. Similar to the skew correction, two steps are applied to remove the slant. First, the slant angle is estimated from the image. Second, a shear transformation with the estimated slant angle brings the handwriting into an upright position. Figure 3.2 illustrates the slant correction.

### 3.1.3 Normalisation of the Writing Regions

A text line can be horizontally divided into three parts, i.e. the descender part, the middle part where the body of the text is located, and the ascender part. During baseline normalisation these parts are scaled to a predefined height. This normalisation makes



**Figure 3.3** Example illustrating the normalisation of the writing regions. The parts of the text line image, i.e. descender, middle, and ascender part are estimated. In this example no descender part is present. The parts are then scaled to predefined height.



**Figure 3.4** Example of width normalisation. A horizontal scaling transformation is applied. The goal is that the average width of the characters becomes similar for any handwriting.

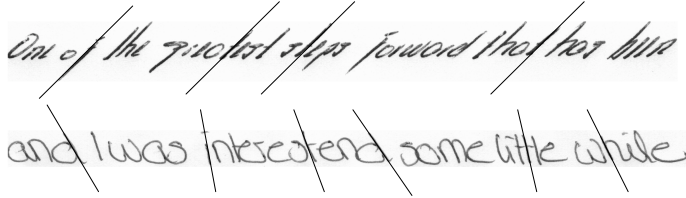
the feature extraction more robust because the location of the handwriting is correlated to the corresponding characters. Some characters, e.g. ‘a’ or ‘e’, are entirely written in the middle part, whereas uppercase letters and characters like ‘l’ and ‘t’ also involve the ascender part.

### 3.1.4 Width Normalisation

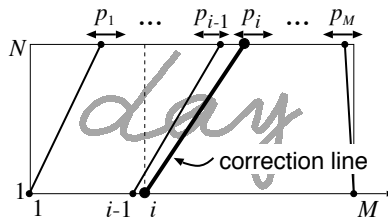
Because the width of the characters typically differs from writer to writer, a width normalisation step is applied. The number of characters in a text line is estimated by counting the number of black-white transitions. The width of the characters is then normalised using a horizontal scaling transformation which is based on the estimated number of characters and a predefined average character width.

### 3.1.5 Non-Uniform Slant Correction

In the conventional slant correction technique described in Sect. 3.1.2, the average slant angle is estimated and uniformly corrected by a single shear transformation. This technique performs well under the assumption that the text line is written with a constant slant. However, in some handwriting styles the slant angle fluctuates not only between different words, but also within a single word. An example is given in Fig. 3.5 where the first line shows a handwriting with a constant slant angle, and the slant angle of the second handwriting fluctuates. The existence of such non-uniformly slanted handwriting styles is the motivation for a local estimation of slant angles and the corresponding



**Figure 3.5** Example of slanted handwriting. The first line shows a uniform slant, whereas the slant angles of the second line are non-uniform.

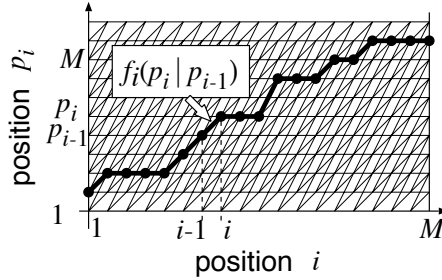


**Figure 3.6** Representation of non-uniform slant angles by correction lines.

non-uniform correction step. The non-uniform slant correction applied in this thesis to handwritten text lines was introduced in [127] for isolated words. The slant correction problem is formulated as a global estimation problem of local slant angles. The sequence that maximises the objective function is determined by a dynamic programming algorithm.

The non-uniform slant correction is applied as follows. Assuming an  $M \times N$  text line image, the problem of the non-uniform slant correction is equivalent to the problem of estimating the local slant angle at each horizontal position  $i \in [1, M]$ . To represent the local slant angle at position  $i$ , a line segment from  $(i, 1)$  to  $(p_i, N)$  is used, where  $p_i$  denotes the horizontal position of the upper-end of the line segment. This line segment is called the  $i$ th correction line. Figure 3.6 shows several correction lines. The slope of the  $i$ th correction line expresses the local slant at position  $i$ .

Consequently, the estimation of local slant angles can be treated as a problem of optimising the slope of the correction line, represented by  $p_i$ , at position  $i = 1, \dots, M$ . The observation of text lines for defining an optimisation criterion reveals that the local slant angles have two properties. First, long near-vertical strokes tend to exhibit local slant angles clearly, and second, left-to-right transitions of the local slant angles



**Figure 3.7** The optimal path problem representing the optimisation problem of  $p_1, \dots, p_i, \dots, p_M$ .

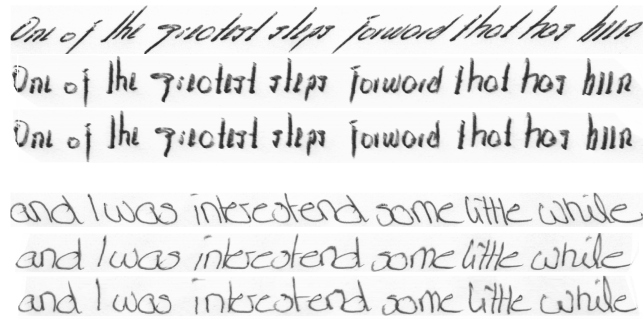
are smooth. Accordingly, the non-uniform slant correction procedure maximises the following function by dynamic programming:

$$F(p_1, \dots, p_i, \dots, p_M) = \sum_{i=1}^M f_i(p_i | p_{i-1}) \quad (3.1)$$

where  $f_i(p_i | p_{i-1})$  is a function to evaluate the “goodness” of the  $i$ th correction line specified by  $p_i$ . The function  $f_i(p_i | p_{i-1})$  is defined as a weighted sum of three functions:  $s_i(p_i)$ ,  $\gamma(p_i | p_{i-1})$ , and  $c_i(p_i)$ . The first function  $s_i(p_i)$  becomes larger if a longer stroke is on the correction line. The second function  $\gamma(p_i | p_{i-1})$  evaluates the similarity between the slopes of the  $i$ th and the  $(i-1)$ th correction line smoothing the transition of local slant angles. The last function  $c_i(p_i)$  is introduced to avoid over-correction, e.g. caused by characters such as ‘X’ and ‘y’ that contain long near vertical strokes which do not exhibit slant angles. The function  $c_i(p_i)$  evaluates the similarity between the local slant angle and an average slant angle in a neighbourhood around the  $i$ th correction line.

The optimisation problem of  $p_1, \dots, p_i, \dots, p_M$  can be treated as an optimal path problem as shown in Fig. 3.7. The path starts from  $i=1$  and ends in  $i=M$  on the  $i-p_i$  search graph and passes through  $(i, p_i)$  next to  $(i-1, p_{i-1})$  on the search graph using the gain  $f_i(p_i | p_{i-1})$ . The sum of the gains along the path becomes the objective function of the optimal path problem. This problem can be solved recursively by dynamic programming. The slant-corrected text line image is then finally obtained by mapping pixels on the correction line between  $(i, 1)$  and  $(p_i, N)$  linearly onto the vertical line between  $(i, 1)$  and  $(i, N)$ .

Figure 3.8 reports the result of uniform and non-uniform slant corrections for the examples shown in Fig. 3.5. In the first example, the difference between uniform and



**Figure 3.8** *Examples of the two different slant correction algorithms. The first line of each example shows the original text line image. The second line shows the image normalised with uniform slant correction, whereas the non-uniform slant correction has been additionally applied to obtain the third line.*

non-uniform slant correction is marginal because the subject wrote with a constant slant angle. In the second example, where the slant angle varies, the non-uniform slant correction provides the visually better correction result.

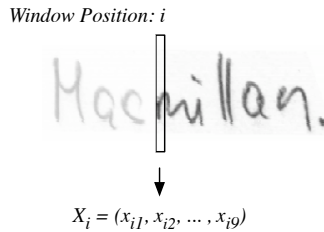
## 3.2 Feature Extraction

After the image normalisation steps, a handwritten text line is converted into a sequence of feature vectors. This sequence of feature vectors is used by the hidden Markov models. To obtain the feature vector sequence, the so called “sliding window” approach is used which is a standard method to extract feature vector sequences from images.

A window of predefined width moves over an image of handwritten text. At each position  $i$  of the window, the feature extraction process extracts a feature vector  $X_i$  based on the image content in the window. Figure 3.9 illustrates the sliding window approach.

For the recogniser used in this thesis, the window has a width of one pixel and the height of the image. It moves from left to right over the handwriting one pixel at a time. Nine geometrical features are extracted at each position  $i$  to build the feature vector  $X_i = (x_{i1}, \dots, x_{i9})$ . The first three features contain the number of foreground pixels in the window as well as the first and the second order moment of the foreground pixels. Features four to seven contain the position of the upper and the lower contour, and the first order derivative from the upper and the lower contour, respectively. The last two features contain the number of vertical black-white transitions and the pixel density between the upper and the lower contour.





**Figure 3.9** With the sliding window approach a feature vector  $X_i$  is extracted at each position  $i$ . Nine geometrical quantities constitute the features of  $X_i$ .

The entire feature vector sequence  $X$  for a handwritten text line image is given by  $X = (X_1, \dots, X_n)$ , where  $n$  is the pixel-width of the image.

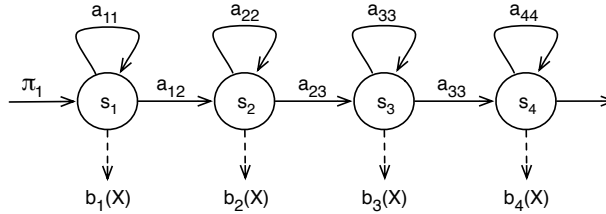
### 3.3 Hidden Markov Modelling

*Hidden Markov Models* (HMMs) are a statistical framework to model observation sequences. HMMs are based on a mathematically solid foundation and have been shown to provide useful models for several real-world processes that produce observation sequences, including biological sequence analysis [29], speech recognition [104], and handwriting recognition [19]. Two main advantages render HMMs especially useful for offline handwritten text line recognition. First, HMMs allow one to perform the recognition and the segmentation at the same time avoiding explicit segmentation. Second, a segmentation of the training data into words and characters is not required because HMMs can be trained using global data, i.e. the image data and the corresponding label word sequence.

#### 3.3.1 Fundamentals of Hidden Markov Models

An HMM  $\lambda$  is defined by a set of states, state transition probabilities, an initial state probability distribution, a set of output values, and emission probabilities.

A finite set of states  $S = \{s_1, \dots, s_n\}$ , a state transition probability matrix  $A = \{a_{i,j} | i, j = 1, \dots, n\}$  where  $a_{i,j}$  is the probability that  $s_j$  follows  $s_i$ , and the initial state probability distribution  $\pi = \{\pi_1, \dots, \pi_n\}$  where  $\pi_i$  is the probability to start in state  $s_i$  describe the topology of an HMM. A large number of topologies can be applied, but many HMM-based applications use a simple linear topology [11, 31, 47, 136], i.e. only transitions to the current state itself and to the next state are allowed. The linear topology has the advantage that only a single parameter, i.e. the number of states, has to be determined



**Figure 3.10** Example of an HMM with linear topology.

to describe the entire HMM topology. Additionally, the number of probabilities to be estimated during training is rather low.

A set  $V$  of possible observation values and the state-dependent emission probability distribution  $b_i(X)$  for each sequence of observations  $X \in V^*$  describe the output characteristics of an HMM. Depending on the nature of observation values, the emission probabilities are either provided by a discrete probability distribution or a continuous probability density functions. If an HMM models a sequence of discrete symbols, it is called a discrete HMM. On the other hand, if an HMM models a sequence of continuous feature vectors, as considered in this thesis, it is called a continuous density HMM. The probability density functions are often modelled with a simple multivariate Gaussian density [89] or a mixture of Gaussian densities [136, 153]; the latter often performs better than a single multivariate Gaussian.

An example of an HMM with linear left-to-right topology is shown in Fig. 3.10. The probability  $\pi_1$  to start in state  $s_1$  is one. The transition probabilities  $a_{i,j}$  are zero for  $i > j$  or  $j > i + 1$ . The example consists of four emitting states. The transitions  $a_{i,j}$  with a priori non-zero probabilities are indicated with an arrow. Only transitions to the state  $s_i$  itself and to the next state  $s_{i+1}$  are allowed. The emission probabilities  $b_i(X)$  are indicated with a dashed arrow.

Three basic problems arise when HMMs are used [104]. First, given an HMM  $\lambda$  and an observation sequence  $X = (X_1, \dots, X_n)$ , how do we calculate  $p(X|\lambda)$ ? Second, given an HMM  $\lambda$  and an observation sequence  $X = (X_1, \dots, X_n)$ , how do we determine a state sequence  $Q = (q_1, \dots, q_n)$  that best explains the observations? Third, how can the parameters of  $\lambda$  be adjusted to maximise  $p(X|\lambda)$ ? The first problem is called Evaluation Problem and can be viewed as scoring how well a given model matches a given observation. It has been efficiently solved with an algorithm called forward-backward procedure [4]. The second problem attempts to uncover the hidden part of the HMM by detecting the optimal state sequence, a problem that, in general, has no unique solution. It has been solved by the Viterbi decoding algorithm [34, 137]. The third problem must

be addressed to train an HMM and is crucial for most applications because it allows the optimal adaptation of model parameters to observed training data. The problem has been solved by the Baum-Welch algorithm [5]. This algorithm iteratively optimises the parameters of an HMM and is a special instance of the expectation-maximisation algorithm [26].

### 3.3.2 Hidden Markov Models in Text Recognition

The goal of a handwritten text line recogniser is to identify the most likely word sequence  $\hat{W} = (\hat{w}_1, \dots, \hat{w}_m)$  out of each possible word sequence  $W = (w_1, \dots, w_m)$  for a known observation sequence  $X = (X_1, \dots, X_l)$  given by the feature extraction process.

$$\hat{W} = \underset{W}{\operatorname{argmax}} p(W|X) \quad (3.2)$$

However, HMMs can only estimate the likelihood  $p(X|W)$  for a given word sequence model  $W$ . Based on Bayes' rule Eq. 3.2 is therefore reformulated.

$$\hat{W} = \underset{W}{\operatorname{argmax}} \frac{p(X|W)p(W)}{p(X)} \quad (3.3)$$

Because the probability of the observation sequence  $p(X)$  is constant for all possible hypotheses  $W$ , it is neglected and the most likely word sequence  $\hat{W}$  is found by

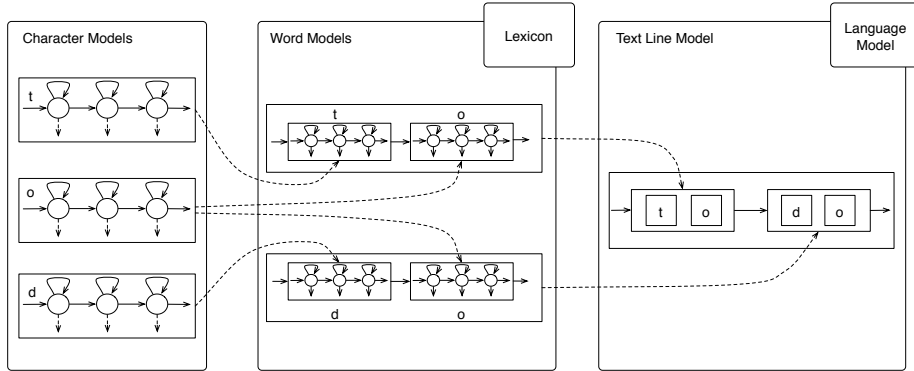
$$\hat{W} = \underset{W}{\operatorname{argmax}} p(X|W)p(W) \quad (3.4)$$

Therefore, the result of the HMM classification must be combined with the word sequence probability  $p(W)$  provided by a statistical language model. Language model issues including the exact integration into the recognition procedure are described in the next chapter.

The following paragraphs discuss some important issues that must be addressed when HMMs are used in handwritten text line recognition.

The first issue to be addressed is the selection of the HMM topology. As discussed above, the simple linear HMM topology has some important advantages over more complex topologies, especially the reduction to a single parameter which defines the number of states. Therefore, most handwriting recognition systems, including the one in this thesis, use a linear topology.

Whereas few published works model each word of the lexicon with an HMM, most current systems model the characters and concatenate these models to word models



**Figure 3.11** Example of building a text line model by concatenating word models which are themselves built from concatenated character models.

based on the lexicon. The advantage of the latter approach is twofold. First, many different words are used to train the same characters providing better models. Second, words that do not occur in the training set can be modelled allowing for the development of large-lexicon tasks. Once the word models are available, a language model is used to build text line models by concatenating word models. An example of a text line model originating from word models which are built by concatenating character models is provided in Fig. 3.11. Note that the same model for character ‘o’ is used in both words, i.e. in *to* as well as in *do*.

The number of states must be defined for each character HMM. Earlier work chooses the same number for each character model [89, 94]. Motivated by the individual average pixel-length of the characters, more recent systems use individual number of states for each character [47, 150]. In this thesis, the Bakis method is applied to find the number of states for each character [2]. The Bakis method determines the number of states for each character HMM based on the average number of observation vectors.

The number of Gaussian mixture components to model the output distribution in each state is another important parameter of an HMM-based recogniser. The optimal number depends on the size and on the nature of the training set [47, 131]. Several methods have been proposed to optimise the number of Gaussian mixture components on a validation set. In the recognisers used in this thesis, the number of Gaussians is not optimised, but heuristically determined based on previous work to reduce computational effort and to avoid overfitting to the validation set.

<i>Transcription Recognition</i>	<i>H</i>	<i>D</i>	<i>I</i>	<i>S</i>	<i>N</i>	<i>Correctness</i>	<i>Accuracy</i>
a b c d a b c d	4	0	0	0	4	100%	100%
a b c d a c c d	3	0	0	1	4	75%	75%
a b c d a c d	3	1	0	0	4	75%	75%
a b c d a d b c d	4	0	1	0	4	100%	75%

**Table 3.1** Evaluation of performance with word level correctness and word level accuracy by hits (*H*), deletions (*D*), insertions (*I*), and substitutions (*S*). All examples assume that the transcription is 'a b c d' against which different hypothetical recognition results are evaluated.

## 3.4 Evaluation Methodology

The evaluation of a text recognition system is more complex than measuring the performance of a single word-classifier. In the latter case, a recognised word is either correct or incorrect. In a text recognition system, a recognised word is either correctly present in the transcription (*H*), a wrongly recognised substitution of a word in the transcription (*S*), or an additionally recognised word, not present in the transcription (*I*). Furthermore, it may occur that a word in the transcription has no corresponding word in the recognition result (*D*).

Two performance measures are usually used in the handwritten text line recognition literature. The first measure is called word level correctness, whereas the second is known as word level accuracy. These two measures are discussed in more detail in the following subsections.

An illustration of the performance measurement is given in Tab. 3.1, where different hypothetical recognition results are tested against the given transcription *a b c d*. The resulting word level correctness and word level accuracy are shown in the last two columns of the table.

### 3.4.1 Word Level Correctness

The word level correctness measures the number of correctly recognised words that are present in the transcription of the handwriting. A correctness of 100% is achieved if all

words that are present in the transcription are correctly recognised. However, additionally recognised words are not taken into account. Thus, 100% word level correctness does not necessarily mean that the recognition result is equal to the transcription. Assuming that the transcription contains  $N$  words out of which  $H$  words are correctly recognised, the word level correctness is defined as follows:

$$\text{Word Level Correctness} = \frac{H}{N} \quad (3.5)$$

The correctness is a suitable performance measure for automatic indexing of handwritten documents. Such indexes can be applied in an information retrieval system to search and browse handwritten documents. However, because additional words are not considered, it is often not a suitable measure for measuring the performance of a transcription system. Especially in the context of multiple recognisers, it can be inappropriate to optimise on the word level correctness. Since additional words are not taken into account, the combination strategy will not perform any decision on which word to use if multiple hypotheses are available, but consecutively output all available hypotheses. This can lead to very undesired recognition results with many words, most of them not present in the transcription.

### 3.4.2 Word Level Accuracy

The word level accuracy takes the insertions into account, and 100% accuracy is only achieved if the recognised word sequence exactly matches the transcription. For this reason, it is usually a better suited performance measure for a transcription system. Assuming  $I$  insertions,  $H$  correctly recognised words, and  $N$  words in the transcription, the word level accuracy is defined as follows:

$$\text{Word Level Accuracy} = \frac{H - I}{N} \quad (3.6)$$

One drawback of the word level accuracy is that there is no lower bound, and it becomes negative if the number of the inserted words exceeds the number of correctly recognised words. However, this is rarely the case in practice.

The word level accuracy is used to measure the performance throughout this thesis. The reason for this decision is twofold. First, the accuracy is considered to be more suited for a transcription task because it does not ignore insertions. Second, the word level correctness can become rather meaningless if the combination module of an ensemble method allows for too many insertions.

### 3.4.3 Statistical Significance Testing

The goal of a statistical significance test is to decide whether a hypothesis is correct or not based on given observations. In the evaluation of handwritten text recognisers, the question is whether the difference between two results is significant enough to state that the new recognition method is better. For this purpose the statistical Z-test is used. The variables to test are the recognition rates at the text line level. Given  $n$  text line results of two methods  $X = (x_1, \dots, x_n)$  and  $Y = (y_1, \dots, y_n)$ , the Z-score is calculated as follows. First, the mean value of results of each method is calculated.

$$\mu_X = \frac{\sum_{i=1}^n x_i}{n} \quad \mu_Y = \frac{\sum_{i=1}^n y_i}{n} \quad (3.7)$$

Next, the variances and the covariance are calculated.

$$\sigma_X^2 = \frac{\sum_{i=1}^n (x_i - \mu_X)^2}{n} \quad \sigma_Y^2 = \frac{\sum_{i=1}^n (y_i - \mu_Y)^2}{n} \quad (3.8)$$

$$Cov(X, Y) = \frac{\sum_{i=1}^n (x_i - \mu_X)(y_i - \mu_Y)}{n} \quad (3.9)$$

Finally, the Z-value is given by:

$$Z = \frac{\sqrt{n} * (\mu_X - \mu_Y)}{\sqrt{\sigma_X^2 + \sigma_Y^2 - Cov(X, Y)}} \quad (3.10)$$

This Z-value is then compared to a Z table that contains the percent of the area under the Gaussian distribution curve between the mean and the Z-value. If the Z value is sufficiently different from the mean value, it is unlikely that the sample mean happened by chance. Typical confidence intervals are 95% ( $Z > 1.65$ ) and 99% ( $Z > 2.33$ ).





# 4

---

## Language Modelling

*Language models* (LMs) attempt to quantify regularities in natural language and to determine the probability of a given sequence of words. This is possible because the position and sequence of words in natural language texts are far from being random. LMs are useful in a variety of applications including speech recognition [23, 59], machine translation [13], spell checking [10, 65], and handwritten text recognition systems [89, 124, 152]. Most LMs that are used in practice, as well as the ones in this thesis, are rather simple  $n$ -gram models, although several improvements including caching, clustering, and skipping models have been proposed.

This chapter is organised as follows. The next section introduces the text corpora used to build the LMs. Fundamental ideas of  $n$ -gram language modelling are described in Sect. 4.2. Next, smoothing is discussed in Sect. 4.3, before the integration of LMs in the HMM decoding step is explored in the last section of this chapter.

### 4.1 English Text Corpora

Large text collections, referred to as corpora, are essential for the construction of lexicons and statistical LMs. The more text is available, the better an LM can quantify the underlying natural language. In this thesis, five different text corpora are used. These are described in the following paragraphs.

The *Brown Corpus of Standard American English* [35] has been collected 1967 by H. Kucera and W. N. Francis at Brown University as a general corpus. It is a carefully compiled selection of American English consisting of about one million words collected from a wide variety of sources. The 500 texts are grouped into 15 categories, e.g. press reportage, general fiction, scientific writings, and humour.

The *Lancaster-Oslo/Bergen Corpus* (LOB) [60] contains printed British English texts. The LOB corpus aims at a general representation of text types to be used in research on different aspects of the English language. The texts have been collected between 1970 and 1978 at the University of Lancaster and the University of Oslo supported by the Norwegian Computing Centre for the Humanities at Bergen. Similar to the Brown corpus, about one million words are present in 500 texts that cover 15 text categories.

The *Wellington Corpus of Written New Zealand English* [3] has been compiled at the University of Wellington in the years 1986-1992. The aim of the Wellington Corpus is to provide samples of written New Zealand English. About one million words in 500 texts are grouped into ten categories.

The *American National Corpus* (ANC) [56] contains a large collection of American English including written texts of all genres and transcripts of spoken data. The corpus has been produced from 1990 onward. The current second release of the ANC provides 13,295 texts containing more than 22 million words.

The largest corpus used in this thesis is the *British National Corpus* (BNC) [16]. The BNC is a 100 million word collection that contains 90% written and 10% spoken language, both from a wide range of sources. The BNC has been built by an industrial/academic consortium in the years 1991-1994 and deals with British English of the late twentieth century.

## 4.2 Fundamentals of $N$ -Gram Language Models

Statistical  $n$ -gram LMs are a simple technique to estimate the probability  $p(W)$  of a word sequence  $W = (w_1, \dots, w_m)$  based on observed word sequences of length  $n$ . The probability that a word  $w_i$  occurs after a given word sequence  $w_{i-n+1}, \dots, w_{i-1}$  is modelled by the  $n$ -gram model. The sequence  $w_{i-n+1}, \dots, w_{i-1}$  is sometimes referred to as history of the word  $w_i$ . The complete formula to estimate the probability  $p(W)$  with  $n$ -gram models is given as follows:

$$p(W) = p(w_1) \prod_{j=2}^{n-1} p(w_j | w_1, \dots, w_{j-1}) \prod_{i=n}^m p(w_i | w_{i-n+1}, \dots, w_{i-1}) \quad (4.1)$$

where the first part corresponds to the case when  $i < n$ . In this case, the probability  $p(w_i | w_{i-n+1}, \dots, w_{i-1})$  is unavailable and the approximation by the corresponding lower order  $n$ -gram is used, e.g.  $p(w_1)$  for  $i = 1$  and  $p(w_2 | w_1)$  for  $i = 2$ .

Note that  $p(W)$  is dependent on  $m$ , the length of the word sequence  $W$ . The probability  $p(W)$  is typically smaller if more words are present in  $W$ .

In practice, bigram ( $n = 2$ ) and trigram ( $n = 3$ ) are used in most current speech and handwriting recognition systems because the estimation of  $p(w_i|w_{i-n+1}, \dots, w_{i-1})$  becomes difficult for large  $n$ . Bigram models assume that the probability depends only on the preceding word  $w_{i-1}$ . Thus, Eq. 4.1 is simplified to:

$$p(W) = p(w_1) \prod_{i=2}^m p(w_i|w_{i-1}) \quad (4.2)$$

Not only the preceding word but the two last words,  $w_{i-1}$  and  $w_{i-2}$ , are taken into account by trigram models:

$$p(W) = p(w_1)p(w_2|w_1) \prod_{i=3}^m p(w_i|w_{i-2}, w_{i-1}) \quad (4.3)$$

Trigram probabilities enable a more precise description of the modelled language. On the other hand, much more text is needed to get a robust estimation of the trigram probabilities.

### 4.3 Smoothing

The  $n$ -gram probabilities  $p(w_i|w_{i-n+1}, \dots, w_{i-1})$  can be estimated using relative frequencies of the  $n$ -grams obtained from large text corpora. In general, however, these relative frequencies are rather unreliable estimations because there are many word sequences  $w_{i-n+1}, \dots, w_i$  that never occur in the training corpus which gives zero probabilities for the corresponding  $n$ -gram estimates. However, a word sequence with a zero probability cannot be recognised (see Eq. 3.4 for details). Therefore, smoothing techniques are typically applied when LMs are built to avoid zero probabilities [44, 113].

The idea of smoothing is to take away some probability from observed occurrences and assign positive values to probabilities of  $n$ -grams that do not sufficiently often occur in the training corpus. To implement this idea, many different techniques have been proposed including Katz smoothing, linear interpolation of models of different order, and many others [20]. In the following subsections, the widely used Katz smoothing and the Kneser-Ney smoothing technique are described. For the sake of simplicity, the smoothing techniques are described for bigram models, however, they can easily be extended to work with higher order  $n$ -grams. All LMs used in this thesis are trained using the Kneser-Ney smoothing technique.

### 4.3.1 Katz Smoothing

The Katz smoothing technique [62] is based on the Good-Turing [43] formula. The underlying assumption is that if a word sequence occurs only a few times in the training corpus, it is probably heavily overestimated.

Assume that  $C(w_{i-1}w_i)$  is the number of times the bigram  $w_{i-1}w_i$  occurs in the training corpus, and let  $n_r$  be the number of bigrams that occur exactly  $r$  times:

$$n_r = |\{w_{i-1}w_i | C(w_{i-1}w_i) = r\}| \quad (4.4)$$

The Good-Turing strategy proposes to discount each  $n$ -gram that occurs  $r$  times and pretend it occurs  $r^*$  times as follows:

$$r^*(r) = (r + 1) \frac{n_{r+1}}{n_r} \quad (4.5)$$

In statistical language modelling,  $r^*$  is almost always smaller than  $r$ . Thus, a certain amount of probability is left-over that can be allocated for unseen bigrams.

Katz smoothing distinguishes between bigrams that occur in the training corpus and bigrams that have never been seen before. If a bigram has been seen in the training corpus the discount function  $r^*$  divided by the counts of the word  $w_i$  is used. For unseen bigrams Katz smoothing backs-off to the unigram distribution multiplied with a normalisation constant  $f$ . The complete Katz smoothing formula is given as follows:

$$p_{katz}(w_i | w_{i-1}) = \begin{cases} \frac{r^*(C(w_{i-1}w_i))}{C(w_i)} & \text{if } C(w_{i-1}w_i) > 0 \\ f(w_{i-1}) \times p_{katz}(w_i) & \text{otherwise} \end{cases} \quad (4.6)$$

where  $C(w_i)$  is the number of times a word  $w_i$  occurs in the training corpus, and  $p_{katz}(w_i)$  is the unigram probability for a word  $w_i$ , which is estimated based on the frequency of occurrence of  $w_i$  in the training corpus.

Katz smoothing is one of the most widely used smoothing techniques, however, it was shown that Kneser-Ney smoothing consistently outperforms Katz smoothing [44].

### 4.3.2 Kneser-Ney Smoothing

The Kneser-Ney smoothing [69] estimates the back-off distribution based on the number of contexts that a word occurs in rather than on the number of occurrences of the word. The discount that is used by Kneser-Ney smoothing is a single discount  $D$ . The number of contexts  $c(w_i)$  that a word  $w_i$  occurs in is given by:

$$c(w_i) = |\{v|C(vw_i) > 0\}| \quad (4.7)$$

where  $C(vw_i)$  is number of times the word sequence  $vw_i$  occurs in the training corpus. Kneser-Ney smoothing is usually used as an “interpolated” model which means that higher order and lower order models are always combined. The formula for interpolated Kneser-Ney smoothing is as follows:

$$p_{kn}(w_i|w_{i-1}) = \frac{C(w_{i-1}w_i) - D}{C(w_{i-1})} + \lambda(w_{i-1}) \frac{c(w_i)}{\sum_w c(w)} \quad (4.8)$$

where  $\lambda(w_i)$  is a normalisation constant such that the probabilities sum of to 1.

An illustrating example of the benefit of the Kneser-Ney technique is the smoothing of the word sequence *San Francisco* [44]. *San Francisco* occurs quite often in English texts. This gives the word *Francisco* a high unigram probability  $p_{katz}(Francisco)$ . Consider now the word sequence *of Francisco* which is assumed not to be seen in the training corpus. Then, the probability assumed by Katz smoothing is rather high, i.e. :

$$p_{katz}(Francisco|of) = f(of) \times p_{katz}(Francisco) \quad (4.9)$$

However, it is very unlikely that *Francisco* occurs after a context other than *San*. Because the number of contexts  $c(Francisco)$  that the word *Francisco* occurs in is small, the Kneser-Ney back-off probability will be small, too, representing the fact that *Francisco* is very unlikely to happen after words other than *San*.

$$p_{kn}(Francisco|of) = \lambda(of) \frac{c(Francisco)}{\sum_w c(w)} \quad (4.10)$$

## 4.4 Language Models and Hidden Markov Model Decoding

The HMM-based recognisers used in this thesis are supported by a statistical bigram LM during decoding. The following reasons make bigrams a reasonable choice. First, the amount of training texts is limited and much less training text is required to train a bigram than a higher order  $n$ -gram LM. Second, the computational cost of the decoding step is substantially higher if trigrams are considered. Third, experimental results reported in previous work show a large improvement in performance when bigram models are included in the recognition process [89, 136, 152], but only little or no further performance gain is achieved with trigram models [135, 151].

#### 4.4.1 Grammar Scale Factor and Word Insertion Penalty

According to Eq. 3.4, not only the HMM providing the likelihood  $p(X|W)$  for a word sequence  $W$  and an observation sequence  $X$ , but also the LM that provides the likelihood  $p(W)$  plays an important role during the Viterbi decoding. If a bigram model is used, Eq. 3.4 can be expressed recursively as follows:

$$\phi(s_i) = \phi(s_{i-1}) + \log p(X_i|w_i) + \log p(w_i|w_{i-1}) \quad (4.11)$$

where  $\phi(s_i)$  is the score for the word sequence  $s_i = (w_1, \dots, w_i)$  and  $p(X_i|w_i)$  is the likelihood of a feature vector sequence  $X_i$  output by the HMM given the word  $w_i$ .

Continuous density HMMs and bigram LMs do not provide true probabilities. To compensate for these deficiencies, a heuristic balancing approach introduces two additional parameters  $\alpha$  and  $\beta$  in Eq. 4.11.

$$\phi(s_i) = \phi(s_{i-1}) + \log p(X_i|w_i) + \alpha \log p(w_i|w_{i-1}) + \beta \quad (4.12)$$

The parameter  $\alpha$  is called grammar scale factor and weights the impact of the LM. The segmentation rate of the recogniser can be controlled by parameter  $\beta$  which is known as word insertion penalty. Both parameters have an unclear statistical meaning and are therefore usually determined experimentally on a validation set [105].

The grammar scale factor  $\alpha \geq 0$  controls the influence of the bigram probabilities against the optical model probabilities that are given by the HMMs. It is worth noting that the optimal value of  $\alpha$  depends on the length of the considered word sequences because, according to Eq. 4.1, the LM probability of a word sequence typically decreases if the length of the word sequence increases [95].

The word insertion penalty  $\beta$  is used to balance the word insertion and the word deletion rate. If  $\beta > 0$ , then sequences with more words are preferred by the recogniser, whereas less words are output if  $\beta < 0$ . Because the optimal value of  $\alpha$  depends on the length of the word sequence,  $\alpha$  and  $\beta$  are not independent and should be optimised jointly.

#### 4.4.2 Recognition Lattices

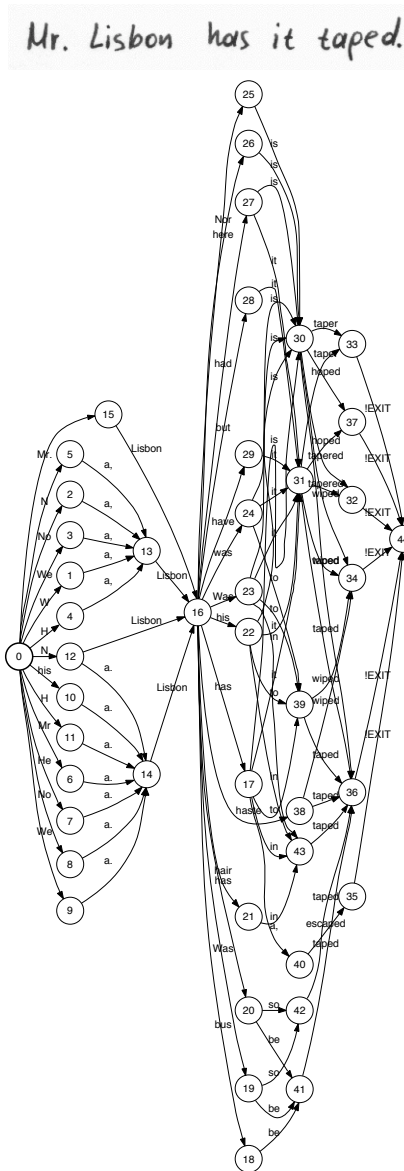
A recognition lattice is a data structure that represents different hypotheses of a handwritten text recogniser in a finite state network. The lattice represents the most promising subspace of recognition results produced by the Viterbi decoding step [145].

A lattice consists of a set of nodes and a set of edges. Each edge represents a hypothesised word between two nodes. The nodes are labelled with a horizontal pixel-position

in the handwriting image. The edges are labelled with a word hypothesis, a likelihood score provided by the HMM, and an LM score provided by the bigram model. All incoming edges of a node must have the same word-label. Accordingly, the corresponding bigram  $(w_i, w_{i-1})$  is directly apparent from a given edge in the lattice.

An example of a lattice produced by a recogniser for the input image *Mr. Lisbon has it taped* is shown in Fig. 4.1. To increase clarity and readability, the original recognition lattice has been pruned, and the scores of the HMMs and the LM are hidden.

Recognition lattices are used for different purposes in this thesis. First, they allow a fast optimisation of the grammar scale factor  $\alpha$  and the word insertion penalty  $\beta$ . The lattices are rescored with different  $(\alpha, \beta)$  values, and thus the expensive Viterbi decoding must be applied only once for each text line in the validation set. Second, in Sect. 5.2, confidence measures are calculated based on alternative candidates that are extracted from the lattice. Third, LM variations produce multiple recognition results from the lattice that are used as ensembles as described in Sect. 8.3. Last, word transition networks are converted to lattices during the LM-based combination method in Sect. 9.3.



**Figure 4.1** Example of a recognition lattice.



# 5

---

## Confidence Measures

A common way to express how certain a recogniser is about its decision is to calculate a confidence measure for each input unit, such as letters, words, or text lines. Based on this confidence measure, rejection strategies can be implemented. If the confidence measure of a letter, word, or text line exceeds a specific threshold, the recognition result is accepted. Otherwise it is rejected. In the context of classifier combination, confidence measures are used to give a result with a higher confidence a higher importance during the combination process.

In previous work, many confidence measures have been proposed. They depend on the application and the underlying recogniser. The following paragraphs survey work in offline handwriting, in online handwriting, and continuous speech recognition.

In offline handwriting recognition, confidence measures for automatic address reading, cheque processing, character classification, and word recognition systems have been proposed. Confidence measures for an HMM-based handwriting recognition system for German address reading are introduced in [8]. To reject isolated handwritten street and city names, four different strategies based on normalised likelihoods and the estimation of posterior probabilities are described. Rejection strategies for cheque processing systems are presented in [45] where an artificial neural network calculates a confidence measure from a set of features. Several confidence measures for an offline handwritten character recognition system are investigated in [100] including recognition score, likelihood ratio, estimated posterior probability, and exponentiated probability. Various rejection strategies for offline handwritten word recognition are proposed in [70] where class-dependent, hypothesis-dependent, as well as a class-independent and hypothesis-independent confidence measures are presented.

In [99], confidence measures are evaluated in the field of online handwriting recognition. These confidence measures are similar to those investigated in [100] for offline recognition. An artificial neural network combining different confidence measures is

used to decide when to reject isolated digits or words. Further confidence measures based on implicit anti-models are investigated in [91]. The confidence measures are integrated in an isolated word recognition system as well as in a sentence recognition system. Four different letter-level confidence measures are applied.

In the field of continuous speech recognition, additional confidence measures that are based on the integration of a statistical LM are used. The integration of the LM in the recognition process can be controlled by two factors: the grammar scale factor and the word insertion penalty (see Sect. 4.4 for details). In [115], the grammar scale factor is used to classify incorrect words in a speech recognition system. Two models based on acoustic stability are presented. Not only the grammar scale factor but also the word insertion penalty is used in [147] in the field of conversational telephone speech recognition. Multiple candidate sentences derived from LM integration variations are used to determine the confidence measure.

The remaining part of this chapter describes the confidence measures used in this thesis. Section 5.1 presents a confidence measure based on normalised likelihoods. Two more elaborate confidence measures based on alternative candidates are described in Sect. 5.2. The last section outlines how the confidence measures are used to implement a rejection strategy.

## 5.1 Likelihood-Based Confidence

The first confidence measure is based on normalised likelihoods. The HMM-based recogniser accumulates likelihoods for each frame, i.e. each position of the sliding window. The resulting likelihood score for a word is used in the decoding step. Because the raw recognition score is influenced by the length of the handwritten word, it is normalised by the number of frames. The result is an average likelihood, which is then used as confidence measure.

The advantage of the likelihood-based confidence measure is that it is simple and fast to calculate. The drawback is that its reliability is restricted because the likelihood scores used by continuous HMMs are far from being real probabilities [101]. Locally, for a given feature vector sequence, the HMMs provide comparable scores that allow for a successful decoding of a text line. However, for different feature vector sequences the scores can often not be compared in a meaningful way.

## 5.2 Confidence Derived from Candidates

More sophisticated confidence measures can be derived if a list of alternative word sequences is available. In addition to the recogniser's top ranked output  $W = (w_1, \dots, w_m)$ ,

the list contains  $K$  alternative candidates  $\hat{W}_1, \dots, \hat{W}_K$  where  $\hat{W}_i = (w_1^i, \dots, w_{m_i}^i)$ ,  $i = 1, \dots, K$ .

The quality of the alternative candidates is a key aspect for a good performance of these confidence measures. In the ideal case, an alternative candidate sequence should distinguish itself from the top ranked output word sequence exactly at the positions where top ranked output words have been recognised incorrectly. Of course, in practice, this is rarely the case as alternative candidates sometimes differ in words that have been recognised correctly or coincide with wrongly recognised words.

### 5.2.1 Generation of Candidates

A common way to produce alternative candidates is the extraction of an  $n$ -best list containing the  $n$  highest ranked transcriptions of a given image of handwritten text. However, it has been shown both in the field of speech [147] and handwriting [148] recognition that candidates based on LM integration variations have the potential to provide better rejection performance than  $n$ -best lists. Therefore, LM integration variations are used in this thesis to obtain alternative candidates.

For an HMM-based recognition system with integrated LM, such as the one used in this thesis, the most likely word sequence  $\hat{W} = (w_1, \dots, w_m)$  for a given observation sequence  $X$  is calculated as follows:

$$\hat{W} = \underset{w}{\operatorname{argmax}} (\log p(X|W) + \alpha \log p(W) + \beta m) \quad (5.1)$$

According to Eq. 5.1, the score  $p(X|W)$  provided by the HMM is combined with the likelihood of a text line  $p(W)$  obtained from the LM. Because the HMM system and the LM merely produce approximations of probabilities, two additional parameters  $\alpha$  and  $\beta$ , which is multiplied with the length  $m$  of the word sequence  $W$ , are required to control the integration of the LM (see Sect. 4.4 for more details).

By varying the two parameters  $\alpha$  and  $\beta$ , multiple candidates can be produced from the same image of a handwritten text. To obtain  $K$  alternative candidates  $\hat{W}_i$ ,  $K$  different parameter pairs  $(\alpha_i, \beta_i)$ ,  $i \in \{1, \dots, K\}$ , are chosen.

It is worth noting that it is not necessary to run the computationally expensive HMM decoding  $K$  times to obtain  $K$  candidates. Instead, a recognition lattice is output by the recogniser and rescored  $K$  times with different  $(\alpha, \beta)$  values to generate the  $K$  alternative candidates.

### 5.2.2 Candidate-Based Confidence Measures

Once the alternative candidates are available, they are each aligned with the top ranked output  $W$  using dynamic string alignment [138]. Based on this alignment, a confidence

Transcription:	Mr.	Lisbon	has	it	taped
Hypothesis:	Mr.	Lisbon	had		escaped
Candidate 1:	Mr.	Lisbon	has	it	taped
Candidate 2:	Mr.	Lisbon	has	it	taped
Candidate 3:	Mr.	Lisbon	had		escaped
$n$ :	3	3	1		1

**Figure 5.1** Counting the number of times a hypothesized word occurs in alternative candidate sequences.

measure  $p(c|w, n)$  is calculated for each word  $w$  of the recognised word sequence  $W$ . The quantity  $p(c|w, n)$  represents the probability of a word  $w$  of the top ranked output being recognised correctly, where  $c \in \{0, 1\}$  (0 stands for incorrect and 1 for correct), and  $n \in 0, \dots, K$  corresponds to the number of times a word  $w$  is observed in the  $K$  alternative candidates. See Fig. 5.1 for an example.

If the training set would be large enough, it would be possible to estimate the probability  $p(c|w, n)$  for every value of  $n$  and all the words  $w$  contained in the lexicon. Since most words occur only a few times (many words occur not at all in the training set) the expression  $p(c|n, w)$  is approximated by two different confidence measures.

In the first approximation Conf1, the probability  $p(c|n, w)$  is estimated by  $p(c|n)$ . The underlying assumption is that the probability of being correctly recognised is independent of the considered word  $w$ . This assumption allows for a straightforward and robust estimation of  $p(c|n)$ . The probability  $p(c|n)$  is then used as a confidence measure Conf1.

$$\text{Conf1} = p(c|n) \quad (5.2)$$

However, the assumption that the probability of a correct recognition is independent of the considered word may be too strong. There are words that are easy to recognise, while others are more difficult. Therefore, the second approximation Conf2 takes into account that some words are more likely to be correctly recognised than others. Conf2 explicitly considers the current word  $w$  instead of assuming that the recognition result is independent of word  $w$ , as is assumed in Conf1. For Conf2 the Bayes' rule is used to reformulate  $p(c|n, w)$ :

$$p(c|n, w) = \frac{p(n|c, w) \cdot p(c|w)}{\sum_{x=0,1} p(n|x, w) \cdot p(x|w)} \quad (5.3)$$

Considering the assumption that  $p(n|c, w) \simeq p(n|c)$ , Eq. 5.3 can be simplified [115]. By this approximation the resulting confidence measure Conf2 is defined as follows:

$$\text{Conf2} = \frac{p(n|c) \cdot p(c|w)}{\sum_{x=0,1} p(n|x) \cdot p(x|w)} \quad (5.4)$$

The probabilities  $p(n|c)$  and  $p(c|w)$  are estimated using relative frequencies obtained from the training set during the training phase. If there are not enough training samples for a word  $w$  to estimate  $p(c|w)$ , confidence measure Conf1 is used instead of Conf2.

### 5.3 Rejection Strategy

Based on the previously described confidence measures, a rejection strategy can easily be implemented. Given a confidence value for each output word, only those words are accepted for which this confidence value exceeds a given threshold  $\tau$ . This threshold controls the strictness of the rejection strategy. If  $\tau$  is increased, more words are rejected, and the accuracy of the remaining words increases. On the other hand, if a smaller  $\tau$  is used, more words are accepted, but the error rate may increase.

To evaluate a rejection strategy the *Receiver Operating Characteristic* (ROC) curve is commonly used that calculates false acceptance and false rejection rates based on a confusion matrix [24, 86]. A word can either be recognised correctly or incorrectly. In both cases the recognition result may be accepted or rejected, which results in one of the following four outcomes:

*Correct Acceptance* (CA) - A correctly recognised word is accepted.

*False Acceptance* (FA) - A word is not recognised correctly but accepted.

*Correct Rejection* (CR) - An incorrectly recognised word is rejected.

*False Rejection* (FR) - A word that is recognised correctly is rejected.

The ROC curve is then constructed by plotting the *False Acceptance Rate* (FAR) against the *False Rejection Rate* (FRR). These error measures are defined as follows:

$$FAR = \frac{FA}{FA + CR} \quad (5.5)$$

$$FRR = \frac{FR}{FR + CA} \quad (5.6)$$

The general goal to be achieved by a rejection strategy is a low FAR and a low FRR at the same time. However, FAR and FRR trade off. Increasing the decision threshold  $\tau$  leads to a lower FAR because less words are accepted. The FRR, however, increases because if more words are rejected, the number of false rejections increases, too. On the other hand, if  $\tau$  is decreased, the FRR also decreases, whereas the FAR increases.



# 6

---

## Experimental Evaluation

This chapter reports the experiments conducted to evaluate the recognition system described in the previous chapters of this part. After the description of the experimental setup in Sect. 6.1, the evaluation of the reference recogniser is described in Sect. 6.2. The experiments with the non-uniform slant correction technique are reported in Sect. 6.3. Next, Sect. 6.4 provides rejection experiments to evaluate different confidence measures. The last section of this chapter discusses the conducted experiments.

### 6.1 Experimental Setup

All experiments reported in this part are conducted on handwritten text lines from the IAM database [90] described in Sect. 2.3. A writer independent task has been considered which means that no information about the writers who contributed to the test set is available during the training and validation phase.

The training set consists of 6,161 text lines written by 283 writers. A total of 56 writers have contributed 920 text lines to the validation set. The test set contains 2,781 text lines produced by 161 writers. A complete listing of the different sets, including the number of text lines, words, characters, and writers is given in Tab. 6.1. The average number of words (characters) in a text line of the test set is 9.1 (46.7).

A total of 81 character HMMs are built, each with an individual number of states. Twelve Gaussian mixture components model the output distribution of each state. Based on the lexicon, word models are built by concatenating character HMMs. These word models and the LM are used to build text line models.

The statistical LM is based on three different corpora, the LOB corpus, the Brown corpus, and the Wellington corpus (refer to Sect. 4.1 for a brief discussion of these

Set	Text Lines	Words	Characters	Writers
Training	6,161	53,841	288,461	283
Validation	920	8,667	43,450	56
Test	2,781	25,414	129,905	161

**Table 6.1** Listing of the number of text lines, words, characters, and writers present in training, validation, and test set.

corpora). A bigram model is built for each of the corpora. The Kneser-Ney smoothing technique is used during the training of the bigram models. These bigram models are then linearly combined with optimised mixture weights [44].

The underlying lexicon consists of the 20,000 most frequent words that occur in the corpora. The lexicon has not been closed over the test set, i.e. there are out-of-vocabulary words in the test set that do not occur among the 20,000 most frequent words included in the lexicon. This scenario is more realistic than a closed lexicon because the texts in the test set are usually unknown in advance. The considered test set contains 6.3% out-of-vocabulary words. This results in a word level accuracy of 93.7% assuming perfect recognition.

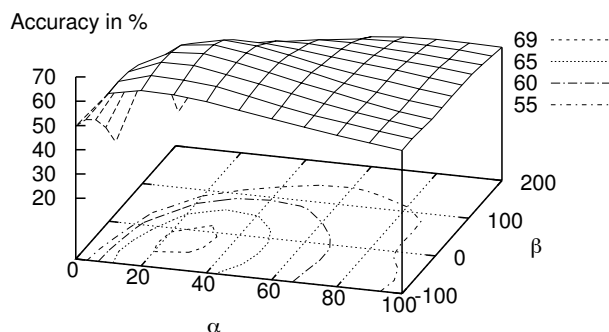
The recognition performance is measured in terms of word level accuracy. This accuracy is defined as the number of correctly recognised words minus the number of insertions, i.e. additionally recognised words, divided by the total number of words in the transcription. A 100% word level accuracy can only be reached if the recognition result matches the transcription word by word (see Sect. 3.4 for more details about performance evaluation).

## 6.2 Reference Recogniser

The reference recognition system is built upon experience of former recognition systems. Many parameters, e.g. the number of states and training iterations, are determined heuristically. Only the integration of the LM is systematically optimised.

Two parameters  $\alpha$  and  $\beta$ , known as grammar scale factor and word insertion penalty, control the integration of the statistical bigram LM into the recognition process (see Sect. 4.4 for more details). The optimisation of these parameters has been shown to substantially improve the performance of a handwritten text recogniser [152]. The systematic optimisation of  $\alpha$  and  $\beta$  is conducted on the validation set as follows: parameter  $\alpha$  is uniformly varied between 0 and 100 (step size: 10), while  $\beta$  is uniformly varied between -100 and 200 (step size: 30). The result of this optimisation is reported in Fig. 6.1. The optimised value for  $(\alpha, \beta)$  is found at  $(30, -10)$  and achieves a word





**Figure 6.1** Optimisation of the integration of the bigram LM on the validation set. The grammar scale factor  $\alpha$  and the word insertion penalty  $\beta$  are systematically varied.

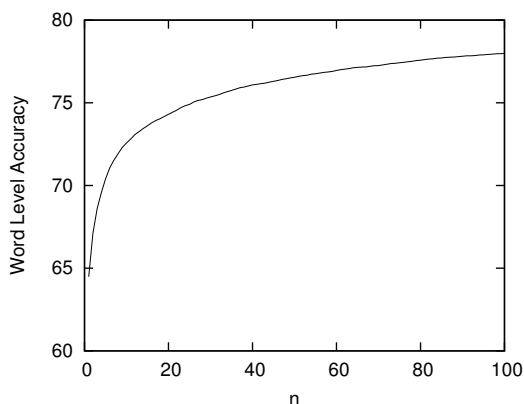
level accuracy of 69.94% on the validation set. This configuration is used as reference recogniser throughout this thesis.

Then, the handwritten text lines in the test set are decoded with the optimised values for  $(\alpha, \beta)$ . On the test set, the reference recogniser achieves an accuracy of 64.48%.

An  $n$ -best analysis on the test set is reported in Fig. 6.2. The  $n$ -best analysis measures not only the performance of the most promising hypothesis, but considers the  $n$ -best candidates, i.e. the result that achieves the highest accuracy among the  $n$ -best candidates is used as recognition result. If not only the most promising hypothesis, but the ten best results are considered, the accuracy increases to 72.57%. For  $n = 100$  the accuracy further increases to 77.98%.

The following paragraphs analyse the recognised word sequences under two aspects: the first aspect is the character length of the recognised word, and the second aspect is the position of a word in the recognised sequence.

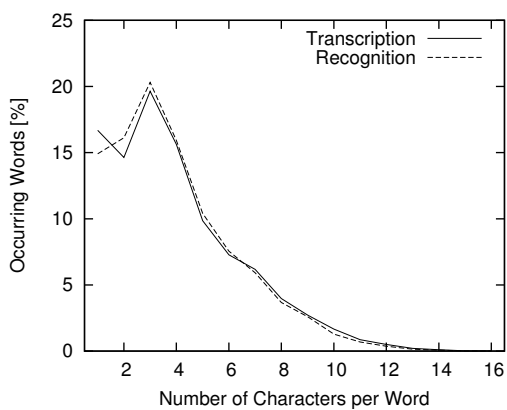
One question that often arises is whether short or long words are more likely to be recognised correctly. A related question is: What is the distribution of the length of the words occurring in the word sequences? The answer to the latter question for the test set used in the experiments described above is shown in Fig. 6.3. The distribution of the length of the words in the transcription as well as in the recognition result is reported. The length of a word is measured by the number of characters in the word.



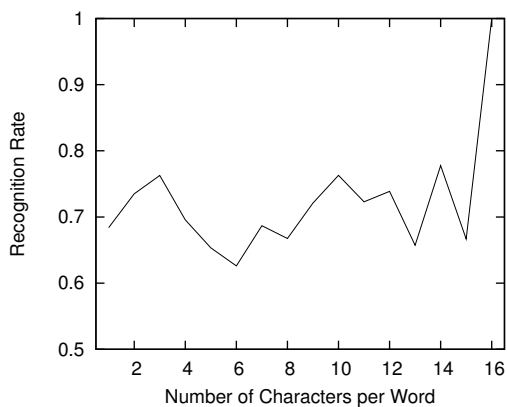
**Figure 6.2** *N*-best results using the hundred best text line candidates on the test set.

About 50% of the occurring words (including punctuation marks) contain three or less characters, and only very few words are longer than ten characters. The longest word in the recognition is *enthusiastically*, which contains 16 characters. Figure 6.4 reports the probability that a recognised word of a given length is correct. Even though many occurring words contain three characters, these words are often recognised correctly. The recognition performance of single-character-words, which are mostly punctuation marks is slightly lower. It is worth noting that this statistic only reports punctuation marks that are recognised; one difficulty, however, is to identify punctuation marks that directly follow a word, which can lead to deletion errors that are not considered in Fig. 6.4. The longest word *enthusiastically* is recognised correctly which leads to a perfect recognition performance for words with 16 characters.

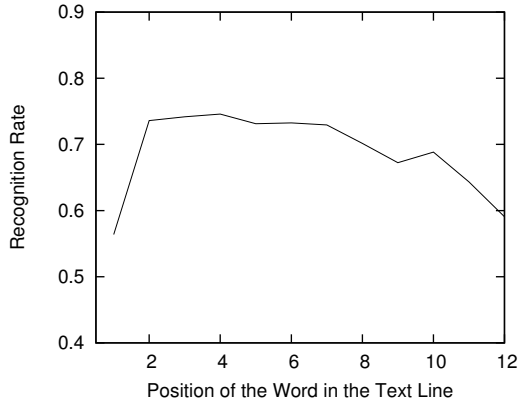
A second interesting question concerns the influence of the LM: How much is the first word negatively affected by the fact that only unigrams and no bigrams apply to the first word? Figure 6.5 reports the probability that a word is recognised correctly when it occurs at a specific position in the recognition result. The recognition rate of words that occur at the first position of the recognised word sequence is about 17% (absolute) lower than words that occur at the second, third, or fourth position. Thus, one can conclude that the fact that bigrams are not applicable to the first word in a sequence has a substantial negative impact on the recognition performance. This problem is intrinsic to the proposed HMM-based text line recogniser with included bigram LM and cannot be avoided. However, if the task changes to recognise entire forms or successive text lines, a recognition system should consider multiple text lines at once in order to avoid this problem for successive text lines.



**Figure 6.3** *Distribution of the length of the recognised words of the test set.*



**Figure 6.4** *Test set word recognition performance depending on the number of characters per recognised word.*



**Figure 6.5** *Test set word recognition performance depending on the position in the recognised word sequence.*

### 6.3 Non-Uniform Slant Correction

To evaluate the effect of the non-uniform slant normalisation technique described in Sect. 3.1.5, three HMM training strategies are considered. In the first training strategy, the HMMs are trained on uniformly slant-corrected handwriting. The second strategy uses non-uniformly slant-corrected handwriting for training. The last training strategy performs a joint training by including uniformly slant-corrected handwriting images as well as non-uniformly slant-corrected handwriting images. Each of the trained recognisers is then validated and tested on uniformly slant-corrected handwriting and non-uniformly slant-corrected handwriting. The integration of the statistical LM is optimised individually on the validation set.

The motivation for the three different training strategies is that the diversity among the character instances in the training set, which is an important issue in writer independent handwriting recognition [132], is higher if no non-uniform slant correction is applied. Thus, training on strongly normalised data probably leads to worse models than training on less normalised data, because in the latter case the variability of handwriting styles is higher which can lead to more general models that perform better on unseen handwriting.

After having optimised the integration of the LM, the results on the validation set are as follows. Whatever training strategy is applied, the non-uniformly slant-corrected handwriting for validation performs better than uniformly slant-corrected handwriting. The best performing configuration on the validation set is joint training with non-uniform

Validation Set		
Training	Validation	
	Uniform	Non-Uniform
Uniform	69.94	70.15
Non-Uniform	68.07	69.56
Uniform & Non-Uniform	71.24	71.85

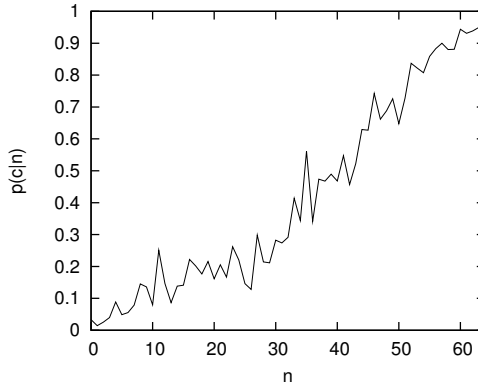
Test Set		
Training	Testing	
	Uniform	Non-Uniform
Uniform	64.48	65.77
Non-Uniform	59.06	62.73
Uniform & Non-Uniform	62.04	65.18

**Table 6.2** Recognition accuracy of the non-uniform slant correction experiments.

slant correction applied to the validation data. This configuration achieves 71.85% word level accuracy.

The results on the test set are similar. If the HMMs are trained on uniformly slant-corrected handwriting, the recognition accuracy increases from 64.48% to 65.77% by the use of non-uniform slant correction on the test data. The recognition rates drop to 59.06% and 62.73%, respectively, if only non-uniformly slant-corrected handwriting is used for training. Contrarily to the validation set, joint training decreases performance to 62.04% accuracy for uniformly slant-corrected test data and to 65.18% for non-uniformly slant-corrected test data. A complete listing of the results on the validation and test set is given in Tab. 6.2.

The conducted experiments show that, whatever training strategy is used, applying non-uniform slant correction to handwriting test data increases recognition performance. This indicates that non-uniform slant correction produces better normalised images than uniform slant correction. However, the non-uniform slant correction did not achieve good performance in the conducted experiments when used to train a writer independent recognition system. Because much writer specific information is removed, the trained recogniser is unable to handle unseen handwriting styles well. In other words, the training of a writer independent recogniser requires enough variability among the training instances to achieve a good performance on unknown handwriting.



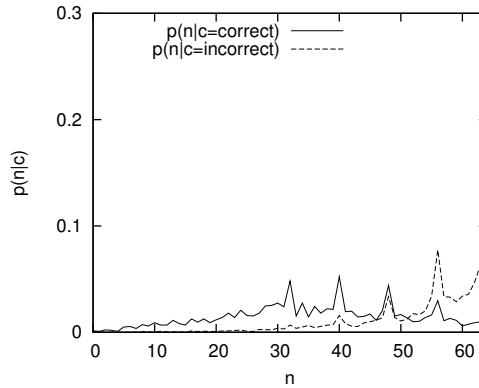
**Figure 6.6** Estimated probabilities  $p(c|n)$  that represent *Conf1*.

## 6.4 Confidence-Based Rejection

This section reports the rejection experiments conducted with the confidence measures described in Chapter 5. To apply the candidates-based confidence measures, several probabilities must be estimated, before an ROC curve can be calculated to evaluate the performance.

First, the probabilities used by Eq. 5.2 and Eq. 5.4 are estimated using relative frequencies on the validation set. Based on preliminary experiments, the number of alternative candidates  $K$  is set to 64. The probability  $p(c|n)$  is estimated by the number of times a word is correct if it occurs  $n$  times in the alternative candidates. If insufficient data is available to estimate  $p(c|n)$  for a given  $n$ , the probability is smoothed to  $n/K$ . The result of this estimation is reported in Fig. 6.6. As expected, the probability that a word is correct increases if  $n$ , the number of times it occurs in the alternative candidates, increases. Analogously, the probability  $p(n|c)$  used by Conf2 is estimated for  $c = \textit{correct}$  and  $c = \textit{incorrect}$ . The estimated probabilities are reported in Fig. 6.7. Last, the probability  $p(c|w)$  that a word is correct is calculated. An extract of the estimated probabilities is shown in Tab. 6.3; e.g. the probability that the word *been* has been recognised correctly is rather high, whereas the word *It* is likely to be recognised incorrectly.

Once the probabilities are estimated on the validation set, the confidence measures are calculated for the words in the test set, and an ROC curve is plotted. The ROC curve describes the performance of a confidence measure by plotting the false acceptance rate against the false rejection rate. A good performing confidence measure achieves a low



**Figure 6.7** Estimated probabilities  $p(n|c)$  that are used by Eq. 5.4 to calculate Conf2.

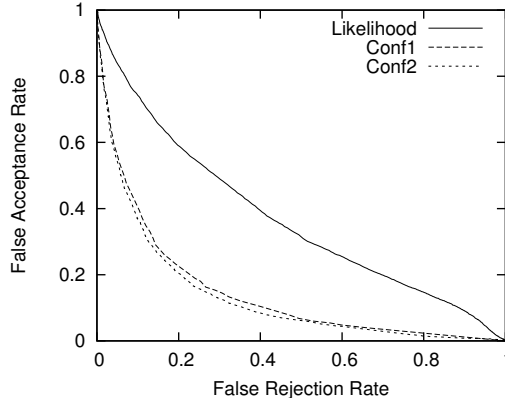
$w$	$p(c w)$
which	0.885
but	0.769
is	0.585
It	0.357
out	0.739
been	0.954
us	0.773
not	0.702

**Table 6.3** Extract of the estimated word probabilities  $p(c|w)$  used by Eq. 5.4 to calculate Conf2.

false acceptance rate as well as a low rejection rate. The ROC curves of the three confidence measures used in this thesis are shown in Fig. 6.8. Both confidence measures based on alternative candidates, i.e. Conf1 and Conf2, perform substantially better than the likelihood-based confidence. Conf2, which takes into account the considered word  $w$ , slightly outperforms the simpler confidence measure Conf1.

## 6.5 Discussion

To conclude the experimental evaluation of the single handwritten text line recognition system, several noteworthy observations and issues are discussed.



**Figure 6.8** ROC curve for the different confidence measures.

One observation that can be made throughout all conducted experiments is that the recognition rate is substantially higher on the validation set than on the test set. Because the LM integration parameters are optimised on the validation set, but, in general, not optimal for the test set, a higher recognition performance on the validation set can be expected. Additionally, it can be assumed that the test contains some “harder” handwritings which are difficult to recognise and which further decrease the test set performance. This problem is well known in handwriting recognition where a rather small number of handwritten text that can substantially increase or decrease the absolute recognition rate of a test set; e.g. in [152], this effect overbalances the optimisations on the validation set, and higher recognition rates are reported on the test set than on the validation set.

The results of the non-uniform slant correction experiments raise the question whether the preprocessing image normalisation should be applied equally to both training and test data. Assuming that the training data perfectly covers the distribution of the test data, applying a worse image normalisation to the training data will lead to worse models. However, in practice, the training data is biased and may become even more biased by a better image normalisation because the variability among the training samples is reduced. Thus, training data which is less normalised may lead to models that generalise better on the test set. On the other hand, the handwriting data in the test set usually becomes more readable the better the image normalisation is, and the recognition performance increases.

The confidence measures based on alternative candidates substantially outperform the likelihood-based confidence measure in the rejection experiments. For several reasons



the likelihood-based confidence measure will nevertheless be considered in the ensemble evaluation experiments. First, the likelihood-based confidence measure is fast and easy to calculate. Second, it avoids overfitting because it does not need any parameter to be optimised on the validation set. Third, if a confidence measure is unreliable for rejection, it is not necessarily unusable in a combination process. Locally, for a given feature vector sequence, the HMMs provide comparable likelihood scores that enable a successful decoding of a text line, which can be useful for combination. However, for different feature vector sequences, as used in a rejection context, these scores can often not be compared meaningfully.

Although the proposed system is quite mature and has been investigated for many years, several open problems remain. First, the system is unable to deal with hyphenations. If a word is split into two parts, each on a separate line, the system will most likely produce an error. A possible solution could be to check the endings and beginnings of lines in order to identify word fragments which are then classified with a specific recogniser. Another open problem is the identification and recognition of out-of-vocabulary words. In the experiments about 6.3% of the words cannot be recognised because they do not occur in the lexicon, e.g. proper nouns and misspelt words. To solve this problem, these words must be identified and submitted to a specific recogniser that is not lexicon-based, e.g. a recogniser that uses character  $n$ -gram models.



# **Part II**

## **Ensemble Methods**



# 7

---

## Introduction

The part describes ensemble methods that were developed to improve the reference recogniser presented in Part I. The idea is to automatically generate multiple recognisers that produce diverse results. By combining these results, a higher recognition rate is often achieved compared to the single reference recogniser.

The use of multiple classifiers in an ensemble to solve a given task has been under research for many years. A brief historical overview is given in [41]. Multiple classifier systems have been investigated not only in the pattern recognition community [49, 58, 76], but also in the machine learning and artificial intelligence community.

From a machine learning point of view, there are three reasons to apply ensemble methods [27]. The first reason is statistical. By averaging the result of multiple classifiers, the recognition performance might not increase, but the risk is reduced that an inadequate single classifier is used. The second reason is computational. Several learning algorithms perform some form of a greedy search that might get stuck in local optima. Multiple classifiers, each running the greedy search from a different starting point, can provide a better approximation of the unknown target function. The third reason is representational. If the classifier space does not cover the optimal classifier, e.g. if linear classifiers are used to solve a non-linear problem, an ensemble can better approximate the optimal classifier.

In the field of artificial intelligence Minsky motivated the use of multiple classifiers as follows [93]:

To solve really hard problems, we'll have to use several different representations... if we only rely on any single, unified scheme, then we'll have no way to recover from failure... It is time to stop arguing over which type of pattern classification technique is best... Instead we should work at a higher level of organisation and discover how to build managerial systems

to exploit the different virtues and evade the different limitations of each of these ways of comparing things.

This introduction to ensemble methods for text recognition is organised as follows. First, the main challenges that have to be addressed when ensemble methods are applied to handwritten text line recognition are discussed. Section 7.2 summarises the state of the art to generate multiple classifiers and to combine the results with a special focus on work conducted in the field of handwriting recognition. Finally, a system overview is presented in Sect. 7.3 that summarises the applied ensemble methods.

The remaining chapters of this part are organised as follows. Ensemble generation and selection methods are described in Chapter 8. Various methods to combine the results of the ensemble members are presented in Chapter 9. Next, Chapter 10 proposes a framework to analyse the diversity of an ensemble. The last chapter of this part reports the experimental evaluation of the introduced ensemble methods.

## 7.1 Challenges

This section describes the most important issues that must be addressed when ensembles of handwritten text line recognisers are investigated.

### 7.1.1 Inducing Diversity

Having multiple classifiers in an ensemble that all provide exactly the same result is not interesting because whatever combination strategy is used, no improvement will happen. Therefore, classifiers in an ensemble must be diverse, making as few common errors as possible. In a regression context, i.e. when multiple regression estimators are combined, an exact relationship between the differences among the individual estimators and the overall ensemble accuracy can be defined. In a classification context, no such intrinsic diversity measure is available [12]. However, previously published work has suggested several diversity measures that correlate with the accuracy of the ensemble.

If multiple recognisers are combined that have been developed separately, e.g. using different feature extraction methods or different system architectures, they usually produce a high diversity. However, if all ensemble members are automatically derived from a single base recogniser, as considered in this thesis, generating a sufficient amount of diversity among the individual ensemble members may become an issue.

### 7.1.2 Combining a Large Number of Classes

Many combination methods depend on the class-labels that are output by the individual classifiers. One popular example is the so-called behaviour knowledge space method [54] where every possible combination of classifier outputs is represented by a cell in a look-up table. The most likely class for each cell is estimated during training. To get robust estimations it is important to have enough training samples available [109].

If the number of classes is large, e.g. recognisers having a 20,000 word lexicon, and the ensemble size is not very small, in the present case up to 24 recognisers, behaviour knowledge space and similar methods are not feasible because not enough training data is available to estimate the required decision results of  $20,000^{24}$  cells of the look-up table. Therefore, mainly voting based approaches that do not consider the class-label itself have been applied to combine classifiers that output many different classes.

### 7.1.3 Combination of Label Sequences

Most existing combination rules are not applicable if each classifier of the ensemble outputs a sequence of class-labels rather than just a single class-label. Such a situation occurs in handwritten text line recognition where the result of each recogniser is a sequence of word classes. It cannot be assumed that the sequences produced by the different recognisers are all of the same length because segmentation errors can happen resulting in some recognisers outputting more words than others. Therefore, some synchronisation mechanism is needed.

A possible solution is to apply dynamic programming techniques to align the individual classifiers' outputs and then locally decide which class-label to choose. However, an optimal alignment of  $n$  word sequences is computationally expensive and a local decision may not be sufficient because a recognised word may be dependent on the preceding word.

## 7.2 State of the Art

In this section, currently available ensemble techniques are summarised. First, methods to automatically generate ensemble members are discussed. Next, several methods that allow for the fusion of label outputs are presented. The last part of this summary focusses on ensemble methods in the domain of handwriting recognition, including character, word, and text line recognition.

### 7.2.1 Ensemble Generation

Most ensemble generation methods either vary the data, the features, or the system architecture. Whereas methods varying the data apply to many different classification problems, altering the system architecture is usually task or implementation dependent. A survey of ensemble generation methods is found in [27].

Many different methods have been proposed to obtain multiple classifiers by supplying the classifier with different training data. These methods work especially well for classifiers that produce substantially different outputs in response to small changes in the training data, e.g. decision tree classifiers or neural networks. The best known among these methods are  $k$ -fold cross validation [73], Bagging [9], and Boosting [36]. A well known method that varies the features to generate multiple classifiers is the random feature subspace method [48]. A comparison of different methods can be found in [129].

Much less research has been conducted on using different classifier architectures or varying parts of the classifiers. Typically, such variations are problem specific and not as generally applicable as the use of different training sets. In [98], the number of hidden neurons are changed to produce multiple artificial neural networks. Multiple classifiers are built upon different feature extraction algorithms in [143].

A wide range of methods have been proposed to select classifiers from a pool of classifiers in order to build an ensemble, a strategy known as “overproduce and choose” paradigm. Overviews are given in [76, 112]. The selection can be based on heuristic rules, on diversity measures, on clustering the classifiers, and on various search strategies. The motivation for overproduce and choose is that current ensemble generation methods may not provide an optimal ensemble. Thus, by applying overproduce and choose, recognition rates often improve at a reduced computation complexity.

To achieve improvements with ensemble methods, the individual ensemble members must be diverse. Intuitively speaking, the members should make as few coincidental errors as possible to enable the ensemble to correct the errors of individual members. Hence, measuring diversity becomes an important tool to predict the performance of an ensemble. Because no intrinsic diversity measure is given for label-output classifiers, many diversity measures have been proposed for such multi-class problems. Surveys are given in [12, 77, 141].

### 7.2.2 Decision Level Combination of Label Outputs

Previous work has proposed many different classifier combination strategies. Surveys are given in [68, 76]. Depending on the information that is provided by the individual classifiers, three types of classifier outputs can be distinguished [143].



The first and most general type is called “abstract level”. Each classifier only outputs the top ranked class-label. For this type, plurality voting can be applied which is amongst the oldest strategies for decision making. The label that occurs most often in the results of the classifiers is voted as the final result. To give better classifiers more power in decision making, weighted voting can be used. More sophisticated decision methods consider the a posteriori probability of a given class using naive Bayes combination [68], the dependencies between the classifiers in the ensemble, a method known as behaviour knowledge space [54], or other trainable decision rules [108].

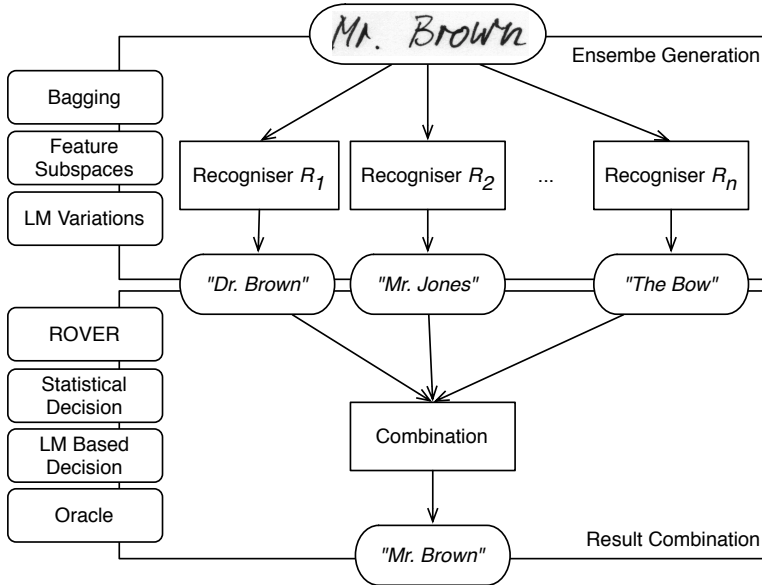
The second type of classifiers is called “rank level” and outputs not only the top ranked class-label but a ranked list of class-labels. For this type Borda count methods apply [6, 50]. The Borda count is calculated for each class by summing up the number of classes ranked below the considered class in the different classifiers. The class with the highest Borda count finally represents the combination result.

Known as “measurement level”, the third type requires the classifiers to output a score for each class-label. These scores can then be combined using product, sum, maximum, minimum, or median rule [68, 75]. The class with the highest value is then regarded as the combination result. More sophisticated trainable approaches use a classifier that decides which class to choose based on the scores of the recognisers.

The combination of handwritten text line recognisers, as considered in this thesis, differs from most other classifier combination problems because the output of the individual recognisers are sequences of class-labels rather than just a single class-label. Standard classifier combination rules, as discussed above, are not directly applicable to the problem of label sequence combination. Because of segmentation errors, it cannot be assumed that the sequences, produced by the different recognisers, all have the same length. Therefore, it has been proposed to use dynamic programming techniques to align the individual outputs of the recognisers before a decision is made locally in the alignment result. However, this topic is still under research, and only a few solutions have been reported in previous studies [33, 87, 88, 117, 140, 144].

### 7.2.3 Ensemble Methods in Handwriting Recognition

In handwriting recognition, several ensemble methods have been presented for digit and character recognition. A survey is given in [106]. An automatic self-configuration scheme that uses genetic algorithms to combine multiple character recognition systems has been proposed in [120]. In numeral recognition, the application of statistical combination methods has been reported in [54]. The behaviour knowledge space methods were especially capable of combining classifiers. A feature selection approach based on a hierarchical algorithm was used in [96] to build ensembles of digit recognisers. In [144], a framework to combine numeral string recognisers was proposed that uses a graph-based approach for combination.



**Figure 7.1** Overview of the ensemble methods applied to handwritten text recognition.

In handwritten word recognition, an evaluation of several decision combination strategies has been reported in [37]. Borda count methods, fuzzy integrals, and multilayer perceptrons have been compared. In [46], various ensemble methods, including Bagging, Boosting, and feature subspace methods have been evaluated.

The investigation of ensemble methods for handwritten text line recognition has started only recently. In [88], a heuristic approach has been used to align and combine multiple handwritten text line recognisers. Positional information of the recognised words is exploited to reduce the search space of the alignment. Three systems that address the task of recognising text written on a whiteboard are combined in [81].

### 7.3 System Overview

Figure 7.1 shows a system overview of the ensemble methods used in this thesis. To apply ensemble methods, two main issues must be addressed. The first issue is how multiple recognisers are obtained from a single base recogniser. The second issue is how the results of the individual recognisers are combined at the decision level.

To generate an ensemble of  $n$  recognisers from a single base recogniser, three different methods are applied in this thesis. The first method is Bagging where multiple recognisers are trained on random bootstrap replicas of the training set. In the second method, known as feature subspace method, different feature subsets are used by the different recognisers. The third method generates different recognisers by varying the integration parameters of the language model into the recognition process.

Each generated recogniser outputs a recognised word sequence. The sequences are then fused at the decision level. In this thesis, four different fusion methods are considered. The first strategy, called ROVER, first aligns the word sequences in a word transition network before a confidence based voting is applied locally at each segment of the network. The second method applies a statistical decision instead of voting to decide which class-label to choose. The third method includes language model information into the combination process. The last combination method is called oracle and calculates a theoretical combination performance assuming a perfect combination rule, to obtain an upper bound on the recognition performance of any combination method.



# 8

---

## Ensemble Generation

To implement an ensemble method, two main issues have to be addressed. First, an ensemble creation strategy must be defined to generate multiple classifiers. The second issue is to find an appropriate combination method to fuse the results of the individual classifiers and to derive the final result. This chapter addresses the ensemble creation problems, whereas the combination methods are discussed in the next chapter.

Basically, there are three different strategies to automatically create multiple diverse classifiers. Under the first strategy, the training data is altered, the second strategy changes the features, while under the third strategy the architectures of the recognisers is varied. This thesis covers one method for each strategy; namely Bagging that alters the training data, the random feature subspace method that varies the features, and LM integration variation where the architecture is altered such that the integration of the underlying statistical LM differs for the different ensemble members. To determine the final ensemble, an overproduce and choose strategy is applied by running a greedy forward search to select the individual ensembles members.

The remaining part of this chapter is organised as follows. Next, the Bagging method that generates ensembles by bootstrapping the training set is described. Section 8.2 presents the random feature subspace method, whereas the ensemble generation method based on variation of the integration of the LM is introduced in Sect. 8.3. The last section of this chapter describes the overproduce and choose strategy to select ensemble members.

### 8.1 Bagging

Bagging is an acronym for *Bootstrap Aggregating* introduced in [9], and it was among the first methods proposed for ensemble creation. The ensemble contains classifiers trained on bootstrap replicas of the training set.

Original Training Set $S$ :	1, 2, 3, 4, 5, 6, 7, 8
Training Set $S_1$ :	2, 7, 8, 3, 7, 6, 3, 1
Training Set $S_2$ :	7, 8, 5, 6, 4, 2, 7, 1
Training Set $S_3$ :	3, 6, 2, 7, 5, 6, 2, 2
Training Set $S_4$ :	4, 5, 1, 4, 6, 4, 3, 8

**Figure 8.1** *Imaginary example of the Bagging method.*

Given a training set  $S$  of size  $N$ , the Bagging method builds  $n$  new training sets  $S_1, \dots, S_n$  each of size  $N$  by randomly choosing elements of the original training set  $S$ . The same element may be chosen multiple times. This method is known as bootstrap sampling [30]. Because the elements are chosen with an equal probability, on average 63.2% of all training elements of  $S$  are covered by a generated training set  $S_i$ .

A classifier  $C_i$  is then trained for each of the generated sets  $S_i$ . Thus, an ensemble of  $n$  diverse classifiers is obtained from the Bagging method. The ensemble members are diverse because they are trained on different training sets. To benefit from different training sets, the base classifier should be unstable in terms of training data variation. Typical unstable classifiers are neural networks and decision trees, whereas  $k$ -nearest neighbour is known as a stable classifier.

A hypothetical example of Bagging is given in Fig. 8.1. The original training set  $S$  contains  $N = 8$  samples. Then,  $n = 4$  bootstrap replicas  $S_1, \dots, S_4$  are generated. Because Bagging resamples the training set with replacements, some samples are represented multiple times in  $S_i$  while other samples are left out. Accordingly,  $S_1$  contains samples 3 and 7 twice, but does not contain either sample 4 or 5. As a result, the recogniser trained on  $S_1$  might obtain a higher error rate than the recogniser using all the training data available in  $S$ . In fact, all four generated Bagging recognisers could result in lower performance. However, when combined, these four recognisers can achieve higher accuracy than that of the single recogniser trained on  $S$ . The diversity among the Bagging recognisers often compensates for the decrease in performance of any individual recogniser.

## 8.2 Feature Subspace

A common way to build ensembles of classifiers is to define subsets of features, either disjoint or overlapping. The feature selection can be performed in several ways. In some classification problems the input features are naturally grouped; e.g. in handwritten digit recognition, an image can be examined from different viewpoints, i.e. pixels

or Fourier coefficients [28]. These groups are then used as feature subspaces. Another common strategy to build feature subsets is to apply random selection [14, 48] which is termed random feature subspace method. The individual recognisers use only a subset of all available features for training and testing. These subsets are chosen randomly with a fixed size  $d$ . The only constraint is that the same subset must not be used twice.

In this thesis the random feature subspace method is used. The underlying handwriting recognition system uses only nine features, which is a rather low number (see Sect. 3.2 for more details on the extracted features). The dimension of the subsets  $d$  is set to six. This number has been found to be optimal for a similar task in the field of handwriting word recognition [46].

The total amount of information that is available to train a recogniser is approximately the same with the feature subspace and with the Bagging method. However, only six out of nine features are used of the test data by a feature subspace recogniser, whereas the Bagging recognisers use information of all nine features extracted from the text lines of the test set.

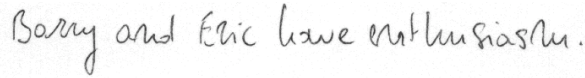
### 8.3 Language Model Variation

One possible architecture modification to create multiple recognition results is to alter the integration of the statistical LM. It has been shown that those parts of a recognised word sequence that are sensitive to changes in the underlying LM are often recognised incorrectly [147, 148]. For these parts alternative interpretations can improve the recognition performance.

For an HMM-based recognition system with integrated LM, the most likely word sequence  $\hat{W} = (w_1, \dots, w_m)$  for a given observation sequence  $X$  is calculated as follows:

$$\hat{W} = \underset{W}{\operatorname{argmax}} \left\{ \log p(X|W) + \alpha \log p(W) + m\beta \right\} \quad (8.1)$$

Equation 8.1 implies that the likelihood of the optical model  $p(X|W)$ , which is the result of the HMM decoding, is combined with the likelihood  $p(W)$  obtained from the LM. Because HMMs as well as LMs only produce approximations of probabilities, two additional parameters  $\alpha$  and  $\beta$  are introduced to control the integration of the LM. The parameter  $\alpha$ , called grammar scale factor, weights the impact of the statistical LM. The term word insertion penalty is used for parameter  $\beta$  which controls the segmentation rate of the recogniser. The word insertion penalty is multiplied with the number of words  $m$  in the word sequence  $W$ . A higher value of  $\beta$  results in more individual words to be output by the recogniser (see Sect 4.4 for more details about grammar scale factor and word insertion penalty).



$\alpha$	$\beta$	Recognition Result
0	-100	Barry arm inch we enthusiasm
0	150	B my arm inch we m run rush :
30	-100	Barry and include enthusiasm
30	150	Barry and Eric have enthusiasm .
60	-100	Barry and include enthusiasm
60	150	Barry and in have enthusiasm .

**Figure 8.2** Multiple recognition results derived from LM integration variation. Specific parameters  $\alpha$  and  $\beta$  are used in Eq. 8.1 to build diverse recognisers.

If the parameters  $\alpha$  and  $\beta$  are altered, multiple recognition results are produced from the same image of a handwritten text line. To obtain  $n$  recognition results,  $n$  different pairs of parameters  $(\alpha_i, \beta_i)$ , where  $i = 1, \dots, n$ , are used.

To reduce computational complexity, the  $n$  recognition results can be obtained from a recognition lattice. These lattices are efficiently rescored with different parameters  $(\alpha_i, \beta_i)$ , where  $i = 1, \dots, n$ .

An example of recognisers based on different integration of an LM is shown in Fig. 8.2. Multiple recognition results are produced for the handwritten text *Barry and Eric have enthusiasm*. This example provides a good illustration of the impact of the two parameters  $\alpha$  and  $\beta$ . If parameter  $\alpha$  increases, the influence of the LM increases and nonsense word sequences, e.g. *B my arm inch we m run rush :*, are eliminated. Furthermore, the influence of parameter  $\beta$  on the segmentation of  $W_i$  can be observed. The average amount of words (including punctuation marks) increases if  $\beta$  increases.

## 8.4 Ensemble Member Selection

To optimise the composition of the ensemble, an ensemble member selection strategy can be applied. The idea is not to use all possibly available recognisers, but only those recognisers that add a benefit to the ensemble. This method is known as “overproduce and choose” [76, 112].

On a validation set, a greedy forward search is applied to find the optimised ensemble. First, the individual recogniser which performs best is selected as the first ensemble member. Then, each of the remaining recognisers is tentatively added to the selected ensemble member, and the performance of the new ensemble is measured. The best



performing ensemble is saved and used for continuation. Iteratively, the best remaining individual recogniser is added to the ensemble. Thus, at each iteration the ensemble size increases by one. This procedure continues until the last available recogniser has been added. Then, the best performing ensemble among all generated ensembles (which, in general, contains not all available recognisers) is determined. This best performing ensemble is used as the final ensemble. Note that with this ensemble member selection strategy, the ensemble size  $n$  is implicitly optimised.

It is worth noting that, because of its greedy nature, the forward search does not provide an optimal ensemble in general. However, given  $m$  recognisers, a complete search would require  $m!$  ensembles to be evaluated which is only feasible for very small values of  $m$ . The proposed greedy forward search substantially reduces the number of ensembles to be evaluated to  $\sum_{k=1}^m k$  and is considered a reasonable trade-off between computational cost and combination accuracy.



# 9

---

## Result Combination

The goal of a decision level combination method is derive the most promising combination result based on a list of results provided by the individual ensemble members. Many different methods for decision level combination of multiple classifiers have been proposed in previous work. Examples include voting by frequency of occurrence, weighted voting, behaviour knowledge space, and Bayes combination [76, 107]. However, as stated in the introduction, the combination of handwritten text line recognisers differs from most other multiple classifier combination problems, because the output of the text line recognisers are sequences of words rather than just a single class. Therefore, standard decision level classifier combination rules are not directly applicable to the problem of combining the results of multiple text line recognisers.

Because segmentation errors may occur in the results of the individual recognisers, it must be assumed that the word sequences differ in the number of words. However, if the word sequence differ in the number of words, existing combination procedures do not apply because the word sequences are not synchronised. To overcome this problem, a two step combination has been proposed [33, 88, 144]. First, the word sequences are synchronised using an alignment procedure. In a second step, the decisions are made based on the result of the synchronisation.

The following combination methods are presented in this chapter. First, a framework called ROVER is described that consists of an alignment and a voting phase. Two extensions that address possible deficiencies in the ROVER framework are proposed in the next sections. The statistical decision method described in Sect. 9.2 implements a trainable decision rule that can benefit from dependencies between individual recognisers. The LM-supported combination described in Sect. 9.3 addresses a major drawback of ROVER which is that all decisions are made individually. The LM-supported combination includes information provided by a statistical  $n$ -gram model to prefer more likely word sequences over less likely ones. The last section provides a concept called

oracle. The oracle calculates a theoretical performance of an ensemble assuming perfect decision and hence provides an upper bound for every decision method.

## 9.1 Receiver Output Voting Error Reduction

In the field of continuous speech recognition, Fiscus addressed the problem of combining the results of five systems [33]. An algorithm called *Recogniser Output Voting Error Reduction* (ROVER) was proposed to analyse and combine the individual outputs. The algorithm was able to significantly reduce the error compared to the best individual recogniser.

The combination based on ROVER can be divided into two phases, alignment and voting. In the first phase, the recognition results are synchronised with an iterative string alignment procedure. In the second phase, a confidence-based voting decision rule extracts the final result at each position of the alignment.

### 9.1.1 Incremental Alignment

Any possible string alignment procedure can be used to synchronise the word sequences of the individual recognisers. However, because the optimal alignment of multiple strings is an *NP*-complete problem [139], a sub-optimal incremental approach performs the alignment. At the beginning, the first two sequences align using a standard string matching algorithm [138]. The result of this alignment is a *Word Transition Network* (WTN). The third word sequence then aligns with this WTN, resulting in a new WTN which next aligns with the fourth word sequence, and so on.

The iterative alignment procedure does not guarantee an optimal solution with minimal edit cost as the alignment is affected by the order in which the word sequences are considered. But in practice, the sub-optimal alignment often provides an adequate solution for the trade-off between computational complexity and alignment accuracy.

An example of multiple sequence alignment using ROVER is shown in Fig. 9.1. Given the image of the handwritten text *the mouth-organ*, the recognisers  $R_1$ ,  $R_2$ , and  $R_3$  produce three different results, none of them providing the correct transcription. In the first step the results of  $R_1$  and  $R_2$  align in a single WTN. Subsequently, the result of  $R_3$  aligns with this WTN. Note that a null transition arc  $\varepsilon$  is added to the WTN because the result of  $R_3$  contains an additional word.

### 9.1.2 Confidence-Based Voting

The voting phase fuses the different word sequences once they are aligned in a WTN. The goal is to identify the best scoring word sequence in the WTN and extract it as the final result.

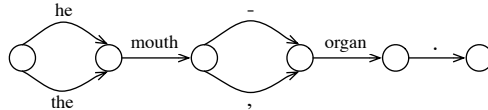
*the mouth-organ.*

$W_1$ : he mouth - organ.

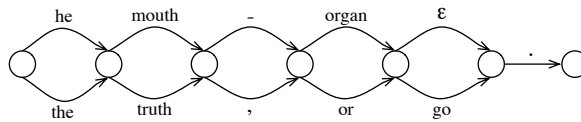
$W_2$ : the mouth, organ.

$W_3$ : the truth - or go.

$W_1 + W_2$ :



$(W_1 + W_2) + W_3$ :



**Figure 9.1** Alignment of multiple recognition results.

The decisions are made individually for each segment of the WTN. Thus, neither of the adjacent segments has an effect on the current segment. The decision depends on the size  $n$  of the ensemble, on the number of occurrences  $m_w$  of a word  $w$  in the current segment, and on the confidence value  $c_w$  of word  $w$ . The confidence value  $c_w$  is defined as the maximum confidence among all occurrences of  $w$  at the current position in the WTN. For each occurring word class  $w$ , the score  $s_w$  are obtained as follows:

$$s_w = \lambda \frac{m_w}{n} + (1 - \lambda)c_w \quad (9.1)$$

The word class  $w$  with the highest score  $s_w$  gives the final result of the current segment of the WTN.

To apply Eq. 9.1, the value of  $\lambda$  is determined experimentally. Parameter  $\lambda$  weights the impact of the number of occurrences against the confidence measure  $c_w$ . Additionally, the confidence measure  $c_\epsilon$  for null transition arcs must be determined because no intrinsic confidence score is associated with a null transition  $\epsilon$ . For this purpose, various values of  $\lambda$  and  $c_\epsilon$  are evaluated on a validation set.

A special case of the confidence-based voting occurs for  $\lambda = 1$  in Eq. 9.1. Then, the

confidence measures  $c_w$  do not have any impact on  $s_w$  which means that simple plurality voting is applied, a method also known as “decision by frequency of occurrence”.

If such a decision by frequency of occurrence is applied to the WTN shown in Fig. 9.1, the combination yields the correct result. Thus, in this example, a perfect transcription is achieved although no individual ensemble member provides the correct result.

## 9.2 Statistical Decision

Previous work has suggested various statistical decision strategies to combine the results of the individual members of a multiple classifier system [54, 108, 143]. The advantage of statistical decision strategies is that the decision making process is trained on existing decisions which can lead to better decision rules than voting or human-driven heuristic rules.

However, if the number of classes is large, most of these methods are not feasible because there is usually not enough training data available to reliably estimate the required probabilities. In contrast, the following statistical decision method handles an arbitrarily large number of classes. It considers not the class-label itself, but which recognisers output a particular class-label. Compared to most previous statistical decision methods, this approach substantially reduces the number of probabilities to be estimated during training. The advantage of the proposed statistical decision method is that it can benefit from dependencies between individual recognisers. The method is used as an extension to the ROVER combination scheme. It uses the same alignment module but applies the novel decision method instead of the confidence-based voting procedure to find the final decision.

### 9.2.1 Methodology

Once the incremental alignment described in Sect. 9.1.1 is complete, the statistical decision method is applied to each segment of the WTN. First, a feature vector  $X_w$  is extracted for each word class  $w$  that occurs in the considered segment. The feature vector contains the confidence measures  $c_w$  of the recognisers that output  $w$ :

$$X_w = (x_{w,R_1}, \dots, x_{w,R_n}) \quad (9.2)$$

where

$$x_{w,R_i} = \begin{cases} c_w & \text{if recogniser } R_i \text{ outputs } w \\ 0 & \text{otherwise} \end{cases} \quad (9.3)$$

The feature vector  $X_w$  is input for a *Multi-Layer Perceptron* (MLP) [97, 111]. The MLP consists of  $l$  input neurons, one hidden layer, and two output neurons. One of the output neurons represents the score for  $w$  being correct, and the other output neuron represents the score for  $w$  being incorrect under input  $X_w$ . The score for correctness estimates the probability  $p(w = \text{correct}|X_w)$ .

The word  $w$  with the highest score for correctness represents the final word class  $\hat{w}$ :

$$\hat{w} = \underset{w}{\operatorname{argmax}} p(w = \text{correct}|X_w). \quad (9.4)$$

This procedure is applied for each segment of the WTN. The final word sequence is then obtained by concatenating the results of all segments of the WTN.

If no confidence measures are available, a simplified version of this method applies. This method uses binary feature vectors as input for the MLP. These feature vectors indicate whether a word is present in the output of a specific recogniser or not. Equation 9.3 is rewritten as follows:

$$x_{w,R_i} = \begin{cases} 1 & \text{if recogniser } R_i \text{ outputs } w \\ 0 & \text{otherwise} \end{cases} \quad (9.5)$$

### 9.2.2 Example

An example of the simplified version of the statistical decision method using Eq. 9.5 is provided in Fig. 9.2. The scanned image of the handwritten text *leave in the autumn* is shown in (a). In this example three different recognisers  $R_1, R_2, R_3$  are considered. The outputs of these recognisers align in a WTN as shown in (b). None of the individual recognisers outputs the correct word sequence.

Next, a binary feature vector is built for each word that occurs in a segment according to Eq. 9.5, resulting in the feature vectors listed in (c). For instance, the feature vector  $X_{is} = (1, 0, 1)$  represents the fact that the word *is* is output by the recognisers  $R_1$  and  $R_3$  but not by  $R_2$ . For each feature vector, the MLP calculates the score for a correct decision. These scores are listed in (d). The final combination result shown in (e) is then derived according to Eq. 9.4 and provides the correct transcription of the input image although none of the individual recognisers produces the correct transcription.

## 9.3 Language Model Decision

If the ROVER algorithm described in Sect. 9.1 is used to combine the results of multiple recognisers, the decisions are made individually for each segment of the WTN

(a) *Input image of handwritten text:*

leave in the autumn

(b) *WTN including the aligned recognition results of  $R_1, R_2, R_3$ :*

	Segment 1	Segment 2	Segment 3	Segment 4
$R_1$ :	leave	is	the	autumn
$R_2$ :	leave	in	that	autumn
$R_3$ :	leave	is	that	august

(c) *Input feature vectors:*

Segment 1:	$X_{leave} = (1, 1, 1)$	
Segment 2:	$X_{is} = (1, 0, 1)$	$X_{in} = (0, 1, 0)$
Segment 3:	$X_{the} = (1, 0, 0)$	$X_{that} = (0, 1, 1)$
Segment 4:	$X_{autumn} = (1, 1, 0)$	$X_{august} = (0, 0, 1)$

(d) *Estimated probabilities for the correctness of a decision:*

Segment 1:	$p(\text{correct}   X_{leave}) = 0.9$	
Segment 2:	$p(\text{correct}   X_{is}) = 0.5$	$p(\text{correct}   X_{in}) = 0.7$
Segment 3:	$p(\text{correct}   X_{the}) = 0.7$	$p(\text{correct}   X_{that}) = 0.3$
Segment 4:	$p(\text{correct}   X_{autumn}) = 0.6$	$p(\text{correct}   X_{august}) = 0.2$

(e) *Combination result:*

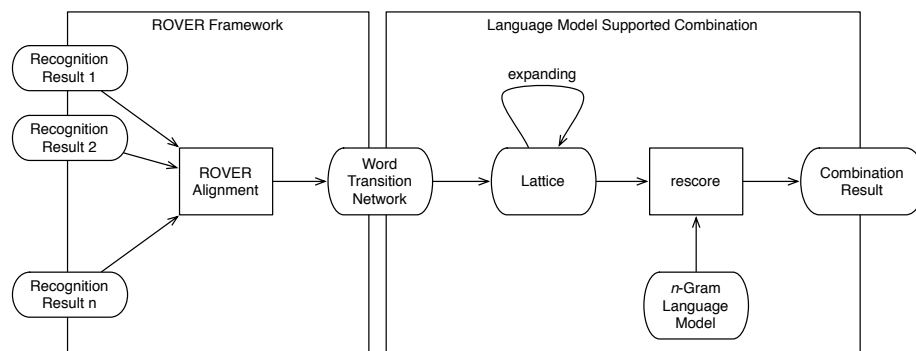
‘leave in the autumn’

**Figure 9.2** *Example of the statistical decision method using binary feature vectors.*

using Eq. 9.1. Thus, none of the adjacent segments has an impact on the decision. Consequently, LM information used by the individual recognisers is lost during the combination procedure.

To overcome this weakness, this section proposes a generic extension to the standard ROVER algorithm that allows for the integration of  $n$ -gram LM information into the combination process. The LM-based combination allows for the preference of more likely word sequences over less likely word sequences. In [117], a similar extension to the ROVER framework is described that uses LM information to break ties during the combination. Ties occur if no single word is able to win the majority voting; i.e. based





**Figure 9.3** Overview of the LM-supported combination procedure.

on frequency of occurrence, no unique decision is possible. However, in contrast to the method proposed in this section, the LM is not used for anything beyond breaking ties in [117].

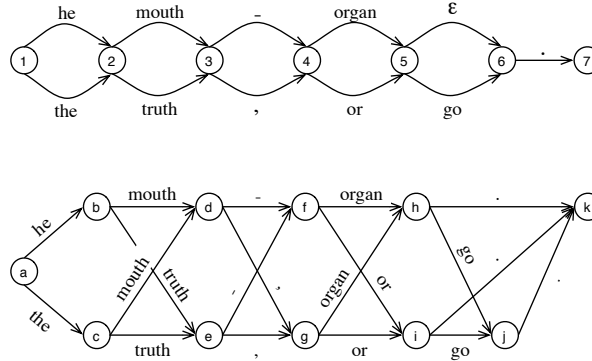
The integration of an  $n$ -gram model into the combination happens in multiple steps. First, the WTN is transformed into a recognition lattice. Then, if necessary, this recognition lattice is expanded before it is rescored with the  $n$ -gram model.

An overview of the combination including an  $n$ -gram model is shown in Fig. 9.3. The  $n$  recognition results are aligned with the ROVER alignment algorithm. The result of this alignment is a WTN. Next, the WTN is transformed into a lattice. If higher order  $n$ -gram LMs are used, this lattice expanded accordingly. Finally, the lattice is rescored using the probabilities provided by the  $n$ -gram language model to build the combination result.

### 9.3.1 From WTN to Lattice

A recognition lattice is a directed graph where all incoming edges of a node have the same word-label  $w$  (see Sect. 4.4.2 for details). This allows for the straightforward integration of a bigram LM in the rescoring process. Therefore, the WTN that is produced by the alignment procedure described in Sect. 9.1.1, is transformed in a lattice.

The transformation of a WTN into a recognition lattice is a particular graph expansion procedure. Beginning with the first segment, the transformation procedure iteratively processes each segment of the WTN. For each edge with a different word-label in the segment a new node is inserted into the lattice. This assures that all incoming edges of a node have the same word-label. The only difficulty occurs when null transition arcs  $\epsilon$  occur in the WTN. Then, the  $\epsilon$ -arcs are recursively contracted and the edges following



**Figure 9.4** Transforming the word transition network of the example of Fig. 9.1 (upper part) into recognition lattice (lower part).

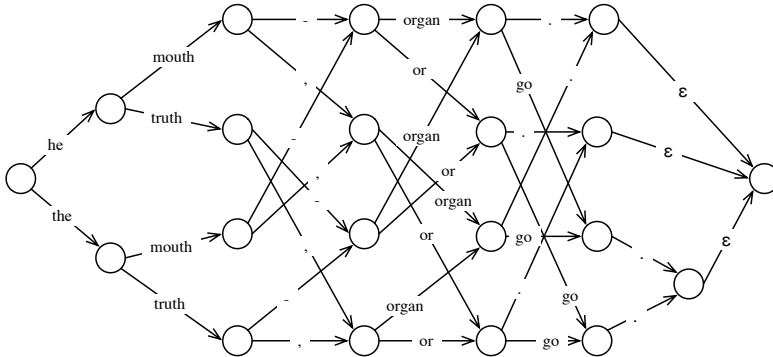
the  $\varepsilon$ -arc are extended, analogously to the  $\varepsilon$ -removal procedure in finite state automata theory [52].

An example of this transformation is given in Fig. 9.4. For each of the two word-labels *he* and *the* in the first segment, we introduce an end node (i.e. nodes 'b' and 'c'). For the second segment of the WTN, containing the word-labels *mouth* and *truth*, the nodes 'd' and 'e' and the corresponding edges are added to the lattice. The procedure continues similarly for the following two segments. Because of the null transition arc  $\varepsilon$  between node '5' and '6' in the WTN, direct arcs, labelled with the dot symbol, from the nodes 'h' and 'i' to node 'k' must be included.

Additionally to the word-label  $w$ , two scores are assigned to each edge of the lattice, i.e. the combination score of Eq. 9.1 and an LM score. The combination score is obtained from the WTN by calculating the score  $s_w$  of Eq. 9.1 for each word  $w$ . If null transition arcs occur in the WTN the scores of the affected edges are multiplied with the score calculated for the null transition arcs during the transformation. The LM score is obtained from the  $n$ -gram model; e.g. for bigram LMs this score is  $p(w_i|w_{i-1})$ .

### 9.3.2 Lattice Expansion

If higher order  $n$ -grams (such as trigram or quadrigram LMs) are used, no straightforward integration of the LM is possible because only the last occurring word is unique in the lattices described above. To allow for a straightforward integration of higher order  $n$ -grams, the lattice must be expanded. Not only all incoming arcs must have the same



**Figure 9.5** Expanding the lattice of Fig. 9.4 to enable rescoring with trigram LMs.

label but the incoming arcs of the  $n - 1$  last nodes. This allows for a simple integration of  $p(w_i|w_{i-n+1}, \dots, w_{i-1})$  as LM score.

Considering a trigram model, Fig. 9.5 shows the expanded lattice for the example of Fig. 9.4. New nodes are introduced into the expanded lattice to fulfil the requirement that the incoming arcs of the last two nodes have the same word-label. This expanded lattice can now easily be rescored with a trigram LM because the incoming nodes of the last two nodes have the same label. Note that  $\epsilon$ -arcs must be included after the last word-labels to obtain one single end node.

### 9.3.3 Rescoring the Lattice

To derive the resulting combination word sequence, the lattice is rescored using the combination scores and the LM scores. The rescoring procedure searches for the path with the highest scores through the lattice. Two parameters are used to weight the influence of the combination scores and the statistical LM. The first parameter  $\mu$  weights the LM score against the combination score, while the second parameter  $\nu$  enables one to control the number of words in the result. The term to optimise during the search procedure is recursively given by:

$$\phi_i = \phi_{i-1} + \log(s_{w_i}) + \mu \log p(w_i|w_{i-n+1}, \dots, w_{i-1}) + \nu \quad (9.6)$$

where  $\phi_0 = 0$ ,  $\phi_i$  is the score at position  $i$  in the lattice, and  $\phi_n$  is the final score for the combination result. The score  $s_{w_i}$  originates from the ROVER combination (see Eq. 9.1), and the probability  $p(w_i|w_{i-n+1}, \dots, w_{i-1})$  is given by the  $n$ -gram model. The parameters  $\mu$  and  $\nu$  must be optimised on a validation set.

Note that this procedure has some similarity to the optimisation of the integration of a statistical LM in the HMM-based recognition described in Sect. 4.4. The difference is that the scores are not obtained from the HMMs, but from the ROVER combination method. Additionally, the search space is typically much smaller, because lattices produced by HMM decoding are usually substantially larger than lattices that originate from ROVER-based WTNs.

## 9.4 Oracle

The oracle decision is a theoretical concept that outputs the correct class-label if at least one ensemble member produces the correct result [50, 76, 142]. The oracle evaluates the hypothetical power of an ensemble of classifiers. Its performance is a theoretical upper bound for any combination method and can be used in comparative experiments because it allows one to quantify the exploitation of a specific combination method given the ensemble members results.

However, it is worth noting that under certain circumstances the oracle performance contains no or little information on the ensemble [108]. For instance, given a two-class problem and an ensemble of two classifiers. If, independent of the input, the first classifier always outputs the first class and the second classifier always outputs the second class, the oracle will achieve perfect recognition.

Similar to most other combination methods, the oracle concept does not apply directly to the combination of handwritten text line recognition because the outputs of the ensemble members are sequences of word classes. To conduct the oracle decisions at the word level in text line recognition, the recognised word sequences are first aligned. For this purpose, the incremental alignment method of Sect. 9.1.1 is used to build a WTN. The oracle decision is then applied to each segment of the WTN to derive the final oracle result. Note that with this strategy, errors made during the alignment can decrease the oracle result. If the correct word is output by a recogniser but not aligned in the correct segment of the WTN, the oracle decision is not able to produce the perfect combination result.

# 10

---

## Diversity Analysis

The goal of ensemble methods is to correct the errors of one ensemble member with the output of other ensemble members. To achieve this goal the ensemble members must be diverse, i.e. the errors of one classifier should be independent of the errors of the other classifiers. Intuitively speaking, the members should make no coincident errors. Although a high diversity of a classifier ensemble does not systematically lead to an improved performance [92], it is considered to be a strong hint towards good performance. Hence, measuring diversity allows one to predict the performance of an ensemble without the need of conducting computationally expensive experiments. Several diversity measures have been proposed for multi-class problems. Surveys can be found in [12, 77, 141].

Two different groups of diversity measures for ensembles of classifiers can be distinguished: pairwise measures and nonpairwise measures. Pairwise measures derive the final diversity of an ensemble of  $n$  classifiers from the  $n(n-1)$  pairwise diversity values. Usually, the mean of these values is used as ensemble diversity. Popular members of this group of diversity measures are correlation, disagreement, and double fault. On the other hand, nonpairwise measures consider all the classifier of an ensemble together and calculate one diversity value directly. Popular nonpairwise diversity measures are entropy and measures based on Sharkey-levels.

Even though appropriate diversity measures for word sequence recognition are potentially useful in the ensemble generation process, no such measures have been published yet. The generic framework presented in this chapter allows one to use existing diversity measures that apply to conventional multi-class classification problems for word sequence recognition problems such as handwritten text line recognition.

Two sections are contained in this chapter. First, existing confidence measures for multi-class problems are described. In the second section, the novel framework is

	$C_i$ correct	$C_i$ wrong
$C_j$ correct	$a$	$b$
$C_j$ wrong	$c$	$d$

**Table 10.1** Probabilities of coincident errors between classifier  $C_i$  and  $C_j$ .

presented that allows one to conduct diversity analysis on ensembles of text line recognisers.

## 10.1 Diversity Measures

Since a high diversity is considered as a strong hint to good ensemble performance, a good diversity measure strongly correlates with the ensemble accuracy. In other words, the goal of a diversity measure is to reliably indicate the performance of an ensemble.

Two groups of diversity measures are used in this thesis. The first group are pairwise measures that derive the ensemble diversity from pairwise diversity values of each pair of ensemble members. The second group are nonpairwise measures that directly calculate the diversity from an ensemble.

### 10.1.1 Pairwise Measures

Pairwise diversity measures consider only a pair of recognisers at a time. Any possible pair of ensemble members produces a diversity value. The average across all pairs gives the final diversity. Based on the probabilities of coincident errors between two recognisers  $C_i$  and  $C_j$ , shown in Tab. 10.1, several measures are calculated.

**Correlation** Because the output of two recognisers can be considered as numerical values (1 for correct, 0 for wrong), the correlation coefficients can be calculated.

$$\rho_{i,j} = \frac{ad - bc}{\sqrt{(a+b)(c+d)(a+c)(b+d)}} \quad (10.1)$$

**Q-Statistic** The Q-statistic [146] for two recognisers is zero if the recognisers are statistically independent. It varies between -1 and 1 and is positive if the recognisers tend to classify the same objects correctly.

$$Q_{i,j} = \frac{ad + bc}{ad - bc} \quad (10.2)$$

**Disagreement** The disagreement measure is a very intuitive measure of diversity [121]. It is the probability that two recognisers will disagree on their decision.

$$D_{i,j} = b + c \quad (10.3)$$

**Double Fault** Another quite intuitive measure is the double fault measure [42] which is the probability that both recognisers make a wrong decision.

$$DF_{i,j} = d \quad (10.4)$$

### 10.1.2 Nonpairwise Measures

The nonpairwise confidence measures applied in this thesis originate from the Sharkey-levels introduced in [119]. Four levels are proposed each describing the occurrence of coincidental errors among the members of an ensemble.

**Level 1** No coincident errors. Each ensemble member produces the correct result.

**Level 2** Some coincident errors, but the majority of the ensemble members provide the correct result.

**Level 3** The majority is not correct but some of the members produce the correct result.

**Level 4** The correct result is not output by any of the ensemble members.

The frequencies of the different levels are used as diversity measures. Thus,  $L_i$  is the frequency that a decision made by the ensemble belongs to level  $i$ , where  $i = 1, \dots, 4$ .

## 10.2 Diversity Analysis for Word Sequence Recognisers

As stated above, the diversity measures of Sect. 10.1 do not apply directly to the considered ensembles of handwritten text line recognisers because not single class-labels are combined but sequences of class-labels. The following framework overcomes this gap by first applying an alignment and labelling process, before existing diversity measures estimate the diversity of the alignment segments.

### 10.2.1 Methodology

Let  $n$  ensemble members have recognised  $n$  word sequences  $(W_1, \dots, W_n)$ . Each of these sequences might contain a different amount of words, and therefore the alignment procedure described in Sect. 9.1 is applied to synchronise the  $n$  word sequences. The result of the alignment is a WTN which consists of  $m$  segments. Each arc in the WTN is labelled with a word out of  $(W_1, \dots, W_n)$ . Null transition arcs  $\epsilon$  occur when the number of words in  $(W_1, \dots, W_n)$  differs.

Next, the ensemble result is derived by applying some decision rule to each segment of the WTN. Any kind of decision strategy can be used, but for the sake of simplicity plurality voting is used. The resulting sequence of decision results constitute the combination result  $\hat{W}$ . Note that if the decision result of a segment is a null transition, this segment does not contribute any word to  $\hat{W}$ .

Once the combination result  $\hat{W}$  and the ground truth are available, the segments of the WTN can be labelled. Therefore, the combination result  $\hat{W}$  first aligns with the ground truth. Based on the alignment the words in the ground truth are mapped to the WTN segments.

Each of the diversity measures described in Sect. 10.1 that were developed for multi-class problems applies now to the segments of the WTN. Averaging the diversities of the segments provides the final ensemble diversity.

### 10.2.2 Example

An example of the entire process of calculating diversity measures for ensembles of handwritten text line recognisers is shown in Fig. 10.1. The input is the handwritten text line *They will be asked to comment* shown in (a). Features are extracted, and each ensemble member performs the recognition step. The recognised word sequences  $(W_1, \dots, W_7)$  are listed in (b).

Next, the word sequences  $(W_1, \dots, W_7)$  are synchronised. An iterative alignment is used for this purpose. Part (c) shows the resulting alignment. Note that null transition arcs  $\epsilon$  are inserted at the beginning of some text lines to align the additional word in  $W_1$  and  $W_6$ . Once the alignment of  $(W_1, \dots, W_7)$  is done, the combination result is calculated for each alignment segment. Plurality voting gets the word sequence *they will be asked to council*. To label the segments, the combination result  $\hat{W}$  aligns with the ground truth  $T$  as shown in (d). Based on this information, the segments of the aligned word sequences of (d) are labelled. The result of the labelling process is shown in (e).

Now, the diversity measures described in Sect. 10.1 that have been originally developed for conventional multi-class classification problems also apply to this example of a sequence recognition problem; e.g. the level 1 diversity measure yields  $4/7$ , and the level 3 diversity measure is equal to  $1/7$ .



(a) Handwritten input text:



(b) Results from the different ensemble members:

$W_1$ : if they will be asked to council  
 $W_2$ : they will be asked to comment  
 $W_3$ : it will be asked to comment  
 $W_4$ : they will be asked to council  
 $W_5$ : they will be asked to council  
 $W_6$ : if it will be asked to comment  
 $W_7$ : they will be asked to council

(c) Alignment of the ensemble results in a WTN:

$W_1$ :	if	they	will	be	asked	to	council
$W_2$ :	$\epsilon$	they	will	be	asked	to	comment
$W_3$ :	$\epsilon$	it	will	be	asked	to	comment
$W_4$ :	$\epsilon$	they	will	be	asked	to	council
$W_5$ :	$\epsilon$	they	will	be	asked	to	council
$W_6$ :	if	it	will	be	asked	to	comment
$W_7$ :	$\epsilon$	they	will	be	asked	to	council

(d) Alignment of the combination result and the ground truth:

$\hat{W}$ :	they	will	be	asked	to	council
T:	They	will	be	asked	to	comment

(e) Labelling the segments:

$W_1$ :	if	they	will	be	asked	to	council
$W_2$ :	$\epsilon$	they	will	be	asked	to	comment
$W_3$ :	$\epsilon$	it	will	be	asked	to	comment
$W_4$ :	$\epsilon$	they	will	be	asked	to	council
$W_5$ :	$\epsilon$	they	will	be	asked	to	council
$W_6$ :	if	it	will	be	asked	to	comment
$W_7$ :	$\epsilon$	they	will	be	asked	to	council
T:	$\epsilon$	They	will	be	asked	to	comment

**Figure 10.1** Example of calculating diversity of ensembles of text recognisers.



# 11

---

## Experimental Evaluation

This chapter reports the experiments conducted with the ensemble methods introduced in the previous chapters of this part. The underlying base recognition system is the one introduced in Part I. The evaluation is conducted on a large set of handwritten text lines from the IAM database. Additional synthetic data has been generated to test the diversity analysis framework.

The next section describes the experimental setup used in all experiments. Section 11.2 reports the experiments conducted using the standard ROVER combination framework. The statistical decision is evaluated in Sect. 11.3, while the experiments with the combination method which includes a statistical LM are described in Sect. 11.4. In Sect. 11.5, the oracle results are given, and the results of the diversity analysis are reported in Sect. 11.6. The last section concludes this chapter by discussing several remarkable issues of the experiments.

### 11.1 Experimental Setup

The experimental setup is similar to the one of the single recogniser experiments described in Sect. 6.1. The same data sets are used. Thus, all input text lines originate from the IAM database [90], and the three data sets, i.e. training (6,161 text lines), validation (920 text lines), and test set (2,781 text lines), provide a writer independent task.

The recogniser evaluated in Sect. 6.2 is used as optimised single reference system. This recogniser is trained on the entire training data. The integration of the statistical LM is optimised as described in Sect. 4.4.

Based on this reference system, 24 ensemble members are generated with each of the ensemble generation methods described in Chapter 8. The number 24 is determined

heuristically. If too few ensemble members are available, the combination method may not be able to increase the accuracy. On the other hand, if too many ensemble members are generated, the computational cost increases substantially, possibly without providing any improvement in the combination accuracy. Thus, 24 Bagging recogniser trained on bootstrap replicas of the training set, 24 random feature subspace recognisers with different feature subsets, and 24 LM integration variation recognisers are generated. The ensembles generated with each single method will be called single-source ensembles. In addition to the single-source ensembles, a multi-source ensemble is created by including all 72 ensemble members from the three ensemble generation methods. The reason for creating multi-source ensembles is that a higher diversity among the ensemble members can be expected if they are generated with different procedures.

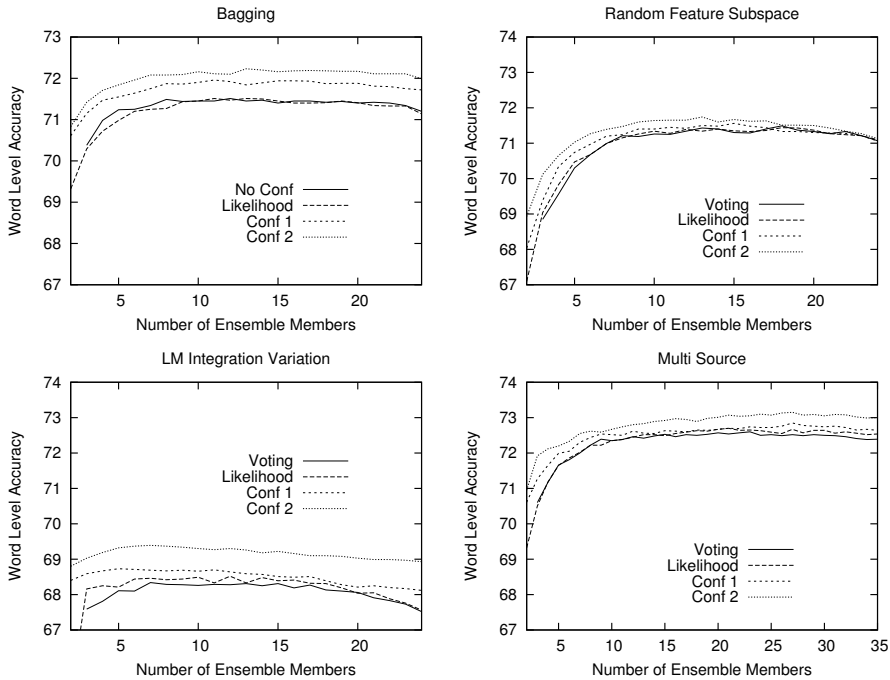
Four different confidence strategies are used. The first and most general strategy does not include any confidence value at all. The second strategy uses the likelihood-based confidence measure described in Sect. 5.1, whereas the third and fourth strategy use the candidate-based confidence measures Conf1 and Conf2 of Sect. 5.2. An ensemble is built with each ensemble generation method, including the multi-source strategy, and with each confidence strategy, which results in a total of 16 ensembles.

Instead of using all available 24, and 72, respectively, recognisers in one large ensemble, the ensemble member selection procedure described in Sect. 8.4 reduces the number of recognisers in an ensemble. The selection procedure is based on a greedy forward search. For computational reasons and to reduce overfitting on the validation set, this selection is conducted with the standard ROVER combination method only. The other combination schemes, i.e. statistical decision, LM-supported combination, and oracle, then use these optimised ensembles.

## 11.2 ROVER Combination

This section provides the experimental evaluation of the combination performed with the standard ROVER algorithm described in Sect. 9.1. After the alignment, the decisions are made for each aligned segment. First, only plurality voting is applied in order to determine the combination result. Then, the three confidence measures, i.e. the likelihood-based confidence and the two candidate-based confidence measures Conf1 and Conf2, are used to conduct confidence-based voting.

Figure 11.1 shows the results of the greedy forward search which is applied on the validation set to select the ensemble members and to determine the ensemble size. During this selection, the parameters  $\lambda$  and  $c_\epsilon$  of the ROVER combination are optimised for each validated ensemble. Consistently, Conf2 outperforms all other confidence measures. If only single-source ensembles are considered, Bagging leads to the best re-



**Figure 11.1** *Overproduce and choose. Validation of the ensemble size and composition with greedy forward search for each ensemble generation strategy and each combination method.*

sults. Including Bagging, feature subspace, and LM integration variation recognisers in multi-source ensembles further increases performance on the validation set.

The optimised ensemble sizes are reported in Tab. 11.1. Ensembles generated with LM variations reach their optimised performance with only 5-12 members. Bagging ensembles have their optimised size at 11-13 members. In contrast, random feature subspace methods require up to 18 ensemble members. As expected, multi-source ensembles have more diversity among the members, thus adding more recognisers to these ensembles benefits the recognition performance.

The results in terms of word level accuracy on validation and test set are given in Tab. 11.2. Similar to the validation set, confidence measure Conf2 outperforms all other confidence measures on the test set. The multi-source ensemble combined with voting achieves a considerably high accuracy of 66.73%, which is an indication that

Ensemble	Voting	Likelihood	Conf1	Conf2
Bagging	12	11	11	13
Random Subspace	18	18	15	13
LM Variations	7	12	5	7
Multi-Source	23	21	27	27

**Table 11.1** *Ensemble size of the different ensemble methods with a given confidence measure. The ensemble size is optimised on the validation set.*

Validation Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	71.51	71.51	71.96	72.23
Subspaces	71.44	71.49	71.56	71.74
LM Variations	68.34	68.52	68.73	69.39
Multi-Source	72.60	72.70	72.85	73.15

Test Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	65.63	65.67	65.88	66.37
Subspaces	65.08	65.32	65.53	65.39
LM Variations	63.38	63.13	63.50	63.83
Multi-Source	66.73	66.50	66.66	67.17

**Table 11.2** *Recognition accuracy of the ROVER combination on validation and test set for different ensemble generation methods and confidence measures.*

the more diverse the recognisers are, the less important is the confidence measure. All improvements over the single reference system that achieves 64.48% accuracy (see Sect. 6.2 for details) are statistically significant at the 5% significance level. If only recognisers generated with the LM variations are used, no improvement over the reference system happens. The best accuracy of 67.17% is achieved by the multi-source ensemble with confidence measure Conf2. If only a single source is used, Bagging with confidence measure Conf2 outperforms all other strategies and achieves 66.37% accuracy.

### 11.3 Statistical Decision

In Sect. 9.2, the statistical decision was introduced. It extends the ROVER framework through a trainable decision method that can handle an arbitrarily large number of

Validation Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	71.78	70.61	71.88	71.94
Subspaces	71.63	71.52	71.54	71.67
LM Variations	68.25	69.00	68.70	69.49
Multi-Source	72.44	72.77	72.67	73.67

Test Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	65.48	64.13	65.68	66.19
Subspaces	65.08	65.31	65.11	65.53
LM Variations	63.48	63.45	63.58	63.96
Multi-Source	66.63	66.69	66.97	67.29

**Table 11.3** *Recognition accuracy of the statistical decision method on validation and test set for different ensemble generation methods and confidence measures.*

classes. The probabilities of the statistical decision are estimated on the validation set by an MLP. The input of the MLP are confidence values output by the different ensemble members. Note that for the ensemble without confidence measures the input for the MLP is a binary vector (see Eq. 9.5 for details).

To avoid over-training of the MLP, the 920 text lines of the validation set are split into training (820 text lines) and validation set (100 text lines). The standard back-propagation algorithm is used to train the weights of the MLP. During the validation phase the number of hidden neurons and the number of training iterations are optimised by minimising the mean square error on the 100 text lines of the validation set.

The results of the statistical decision experiments are provided in Tab. 11.3. Bagging, feature subspace, and multi-source ensembles fused with statistical decision significantly outperform the single reference system (which achieves recognition rates of 69.02% and 64.48% on validation and test set, respectively). Compared to the standard ROVER results in Tab. 11.2, the statistical decision performs better for some ensembles while for other ensembles the standard ROVER gives better results. On the test set, the highest accuracy of 67.29% is achieved with the multi-source ensemble and confidence measure Conf2, which significantly outperforms the ROVER combination at the 5% significance level.

		Validation Set			
Ensemble	Decision	No Conf	Likelihood	Conf1	Conf2
Bagging	ROVER	71.51	71.51	71.96	72.23
	Bigram	71.97	71.93	72.11	72.27
	Trigram	72.17	72.25	72.34	72.45
	Quadrigram	72.16	72.16	72.32	72.47
Subspaces	ROVER	71.44	71.49	71.56	71.74
	Bigram	71.58	-	-	-
	Trigram	71.52	71.57	-	71.80
	Quadrigram	71.58	71.73	-	71.87
LM Variations	ROVER	68.34	68.52	68.73	69.39
	Bigram	68.57	68.93	68.93	-
	Trigram	68.82	69.18	69.18	69.74
	Quadrigram	68.82	69.18	69.17	69.82
Multi-Source	ROVER	72.60	72.70	72.85	73.15
	Bigram	72.75	72.72	-	-
	Trigram	72.80	72.95	-	-
	Quadrigram	72.76	72.87	-	-

**Table 11.4** Recognition accuracy of the LM-based decision on the validation set for different ensemble generation methods and confidence measures. If no result is shown, the best performing decision method is ROVER, i.e. without including an LM.

## 11.4 Language Model Decision

To evaluate the LM-based combination method, bigram ( $n = 2$ ), trigram ( $n = 3$ ), and quadrigram ( $n = 4$ ) LMs are included in the combination process. The bigram LM originates from three corpora, the LOB, the Brown, and the Wellington corpus. A bigram LM is built for each of the corpora. These three bigram models are then linearly combined with optimised mixture weights to build the final LM [44]. To have enough texts to estimate the trigram and quadrigram models, the BNC corpus and the ANC corpus are further considered. Again, for each of the five corpora, an  $n$ -gram LM is built. These models are then combined with optimised mixture weights to build the final  $n$ -gram LM, where  $n \in \{3, 4\}$ .

Before the performance of the LM-based combination is measured on the test set, the parameters  $\mu$  and  $\nu$  of Eq. 9.6 are optimised on the validation set. If the optimal value for parameter  $\mu$  is zero, including the LM into the combination does not provide any benefit on the validation set, and thus no values for  $\mu$  and  $\nu$  are found to measure the performance on the test set.

Table 11.4 reports the result of the LM-based decision method on the validation set.



Ensemble	Decision	Test Set			
		No Conf	Likelihood	Conf1	Conf2
Bagging	ROVER	65.63	65.67	65.88	66.37
	Bigram	66.70	66.57	66.59	66.97
	Trigram	67.10	66.99	66.97	67.35
	Quadrigram	67.10	67.00	67.00	67.41
Subspaces	ROVER	65.08	65.32	65.53	65.39
	Bigram	65.27	-	-	-
	Trigram	65.72	66.33	-	66.12
	Quadrigram	65.78	65.92	-	65.70
LM Variations	ROVER	63.38	63.13	63.50	63.83
	Bigram	63.98	64.22	64.26	-
	Trigram	64.49	64.51	64.52	63.82
	Quadrigram	64.49	64.51	64.33	63.86
Multi-Source	ROVER	66.73	66.50	66.66	67.17
	Bigram	67.22	67.25	-	-
	Trigram	67.50	67.75	-	-
	Quadrigram	67.54	67.82	-	-

**Table 11.5** Recognition accuracy of the LM-based decision on the test set. ROVER acts as a reference system which does not include LM information. If no result is shown, the best performing decision on the validation set is obtained without including an LM.

Although the ROVER combination is highly optimised on the validation set, the LM-based combination method performs better in most experiments, i.e. the optimised value for  $\mu$  is not zero for the reported recognition rates (all other experiments are marked with ‘-’). On the validation set, the best performing experiment achieves 72.95% accuracy and uses multi-source ensembles with likelihood-based confidence measures and a trigram model. However, it does not outperform the standard ROVER method which achieves an accuracy of 73.15% using multi-source ensembles and confidence measure Conf2.

The recognition rates achieved on the test set are reported in Tab. 11.5. What first attracts attention is that each LM-supported combination strategy that achieved an improvement on the validation set performs better than the corresponding ROVER combination on the test set. The best recognition performance is 67.82% achieved with multi-source ensembles, quadrigram LM, and likelihood-based confidence.

Whereas a significant improvement happens when one goes from bigrams to trigrams, the differences in the results of trigrams and quadrigrams are mostly minor. By examining the relative number of  $n$ -grams that can be applied to the transcriptions of the test set, listed in Tab. 11.6, it can be observed that quadrigrams can be used for only

	Bigram	Trigram	Quadrigram
#Out-of-Vocabulary	6.3%	6.3%	6.3%
#Unigrams	21.3%	8.6%	8.6%
#Bigrams	72.4%	38.5%	38.9%
#Trigrams	-	46.6%	30.0%
#Quadrigrams	-	-	16.2%

**Table 11.6** *Relative number of n-grams applicable for the different n-gram models in the transcription of the test set.*

16.2% of the words, whereas all other words that are not out-of-vocabulary must be backed-off to tri-, bi-, or unigrams.

## 11.5 Oracle

This section reports the experiments conducted with the oracle decision described in Sect. 9.4. The oracle decision is a theoretical concept that provides an upper-bound for any combination method given the results of the ensemble members. To obtain the oracle accuracy, the handwritten text lines are first aligned in a WTN. Second, the oracle decision is applied to each segment of the WTN.

The results of the oracle decision are shown in Tab. 11.7. As expected from the ensemble member process and other optimisations, the oracle accuracy is higher on the validation than on the test set. The best performance is achieved with multi-source ensembles. Here the diversity is expected to be higher, and more ensemble members typically lead to better oracle performance. Surprisingly, oracles on random feature subspace ensembles clearly outperform oracles on Bagging ensembles.

Figure 11.2 puts the oracle decision results in context to the ROVER combination results of Sect. 11.2. The results on the validation and the test set look similar despite the lower performance on the test data. In general, there is a tendency that if the oracle performance increases, the performance of the ROVER combination increases, too. This is not true for the subspace ensembles that have a higher oracle performance than Bagging, but the combination accuracy is beneath the accuracy of Bagging ensembles.

One question when working with oracle decision is how much of the optimal performance of the oracle is achieved with a concrete decision strategy. Or in other words, how good is the exploitation of the ensemble? For this purpose, the decision performance is divided by the oracle performance. Figure 11.3 shows the exploitation of the ROVER combination results of Tab. 11.2. All values are between 0.88 and 0.96, which means that about 90% of all decision are made correctly if a correct decision is possible.

Validation Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	76.96	76.79	76.66	76.95
Subspaces	79.19	79.09	78.87	78.59
LM Variations	72.57	73.25	71.25	72.65
Multi-Source	80.06	79.98	80.19	79.95

Test Set				
Ensemble	No Conf	Likelihood	Conf1	Conf2
Bagging	72.29	72.08	71.89	72.51
Subspaces	74.14	74.29	74.05	73.40
LM Variations	68.51	62.21	67.23	68.45
Multi-Source	75.82	75.56	75.78	75.63

**Table 11.7** Recognition accuracy of the oracle decision on validation and test set for different ensemble generation methods and confidence measures.

The LM variation ensembles achieve the best exploitation due to the rather low performance of both oracle and ROVER decision. Candidate-based confidence measures are able to get higher exploitation on similar ensembles. This phenomenon, indicating a useful confidence strategy, not only occurs on the validation set, but generalises on the test set.

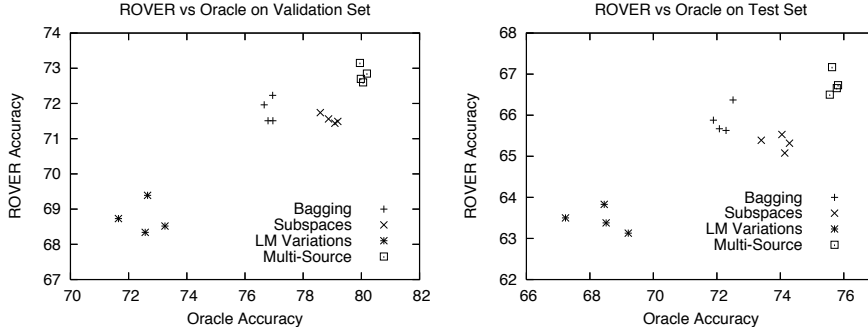
## 11.6 Diversity Analysis Results

Experimental evaluation of the diversity analysis framework proposed in Chapter 10 is conducted on a synthetic and a real-world data set.

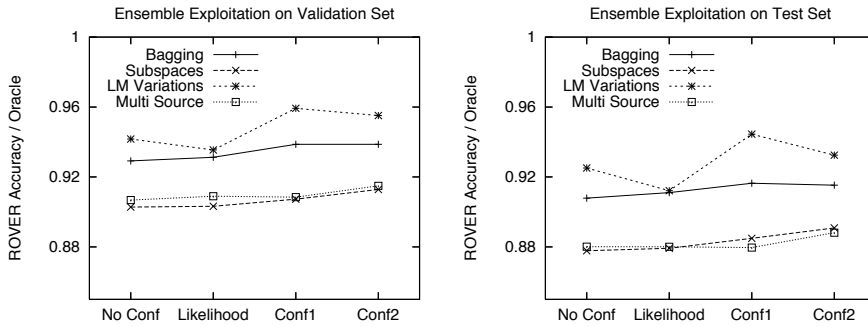
### 11.6.1 Experimental Evaluation on Synthetic Data

The aim of the experiments with synthetic data is to show relationships between the recognition accuracy and the different diversity measures. The advantage of using synthetic data is that the correlation between the ensemble members is under control. Also the number of classes can be chosen to simulate different application domains, e.g. character sequence recognition ( $\sim 80$  classes) or text line recognition ( $\sim 10,000$  classes). Furthermore, an arbitrarily large amount of data can be produced relatively fast. On the other hand, it may happen that the generated ensemble member results do not sufficiently model the results produced by real-word ensemble members.

The diversity analysis framework is evaluated on artificial ensemble results that are generated as follows. First the ground truth is created. To build the ground truth for



**Figure 11.2** Scatterplot of the oracle and the ROVER combination performance on validation and test set.



**Figure 11.3** Dividing the combination accuracy by the oracle accuracy gives a measure of exploitation of the ensemble power. As expected, the exploitation is higher on the validation set.

a word sequence, the length  $n$  of the sequence is first defined by a random number in a given range. Next,  $n$  words are randomly chosen from the underlying lexicon and used as the ground truth word sequence. Second, the ensemble results are generated iteratively. Given the ground truth word sequence, the first ensemble member's result sequence is created randomly with a given accuracy. The next member's result sequence is generated with respect to a given correlation to the previously generated word sequence. This process is continued until the desired number of results is generated.

In the experiments, ensembles that contain five, ten, fifteen, and twenty members with a lexicon of 10,000 word classes are generated. Furthermore, the correlation between two successive members is varied to obtain ensembles with different diversities.

The results for the pairwise measures correlation, Q-statistics, disagreement, and double fault are shown the upper part of Fig. 11.4. All diversity measures clearly correlate to the recognition accuracy using the artificial ensembles. The four different lines that are observed originate from the four different sizes of the analysed ensembles. Thus, the diversity measures are unstable when the ensemble size changes. The lower part of Fig 11.4 shows the results of the nonpairwise level-based diversity measures. The best indicators of a good performance are level 1 and level 4. They are relatively stable against variations in the size of the ensembles.

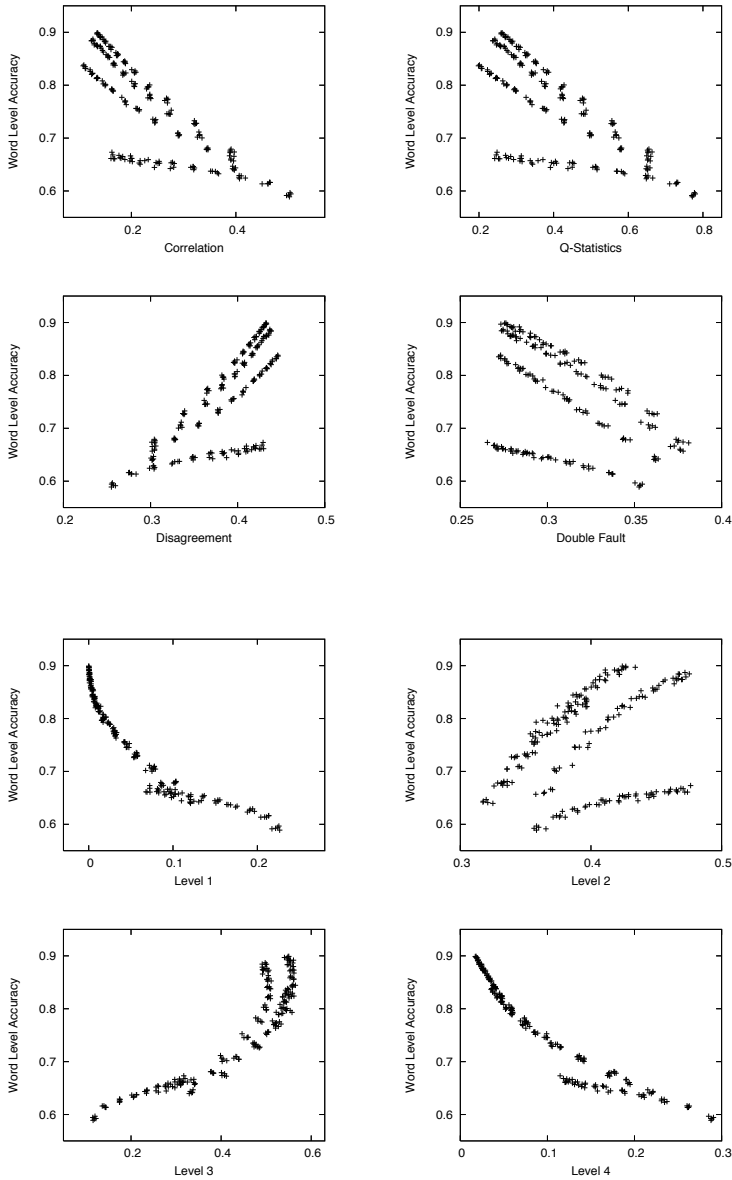
### 11.6.2 Diversity Analysis on Handwriting Data

The experiments conducted to evaluate the diversity analysis framework on real-world ensembles for handwritten text line recognition consider the same eight diversity measures used on artificial data. Four measures, i.e. correlation, Q-statistic, disagreement, and double fault, are pairwise diversity measures, whereas the other four measures are nonpairwise diversity measures based on Sharkey-levels.

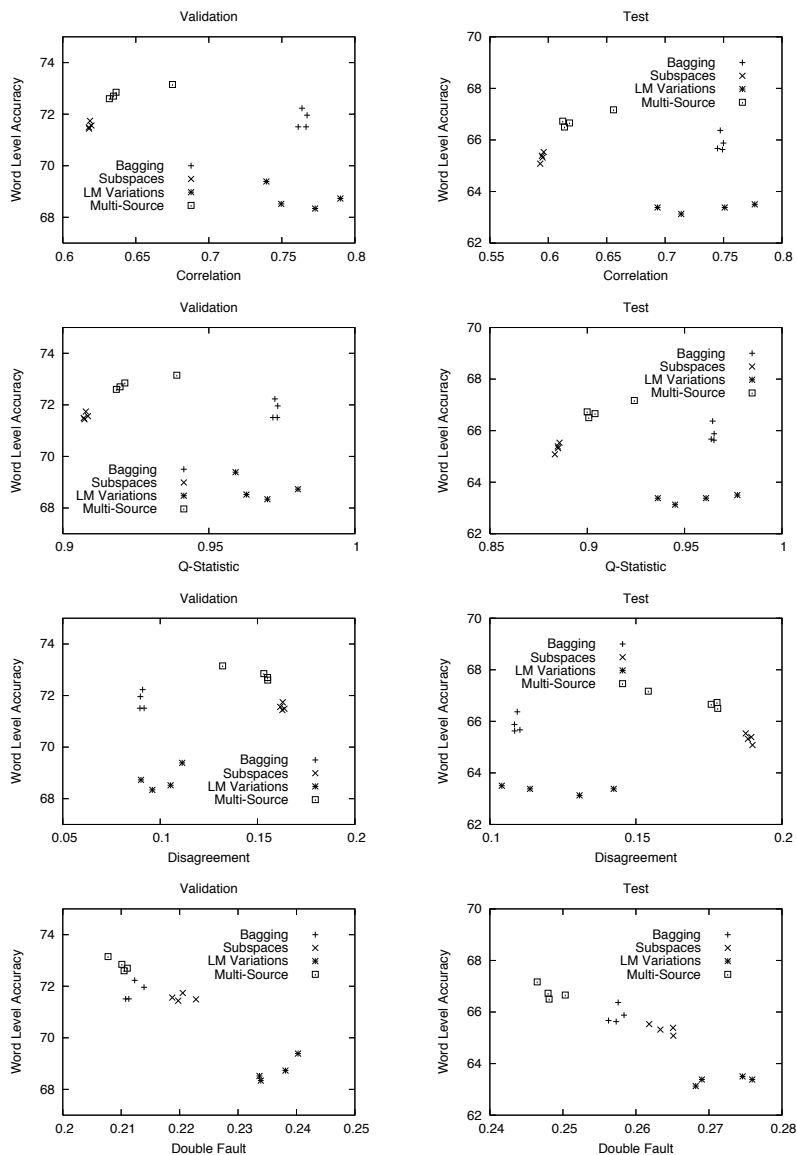
The results of the pairwise diversity measures are shown in Fig. 11.5. Each point in a plot represents an ensemble. The ensembles originate from the different sources, i.e. Bagging, random feature subspace, LM integration variation, and ensembles from all these source referred to as multi-source ensembles. For each source, four confidence strategies are used, i.e. no confidence, likelihood-based confidence, Conf1, and Conf2. This results in sixteen different ensembles. The diversity is plotted against the accuracy of the ROVER combination results reported in Sect. 11.2. The results on the validation set are provided in the left column, whereas the test set results are shown on the right side. The diversity measure with the strongest correlation on both validation and test set is the double fault measure.

The results of the nonpairwise level-based diversity measures are given in Fig. 11.6. Again sixteen ensembles are used to evaluate the relationship between a specific diversity measure and the recognition accuracy of the ensemble. The best results are achieved with the level 4 diversity measure.

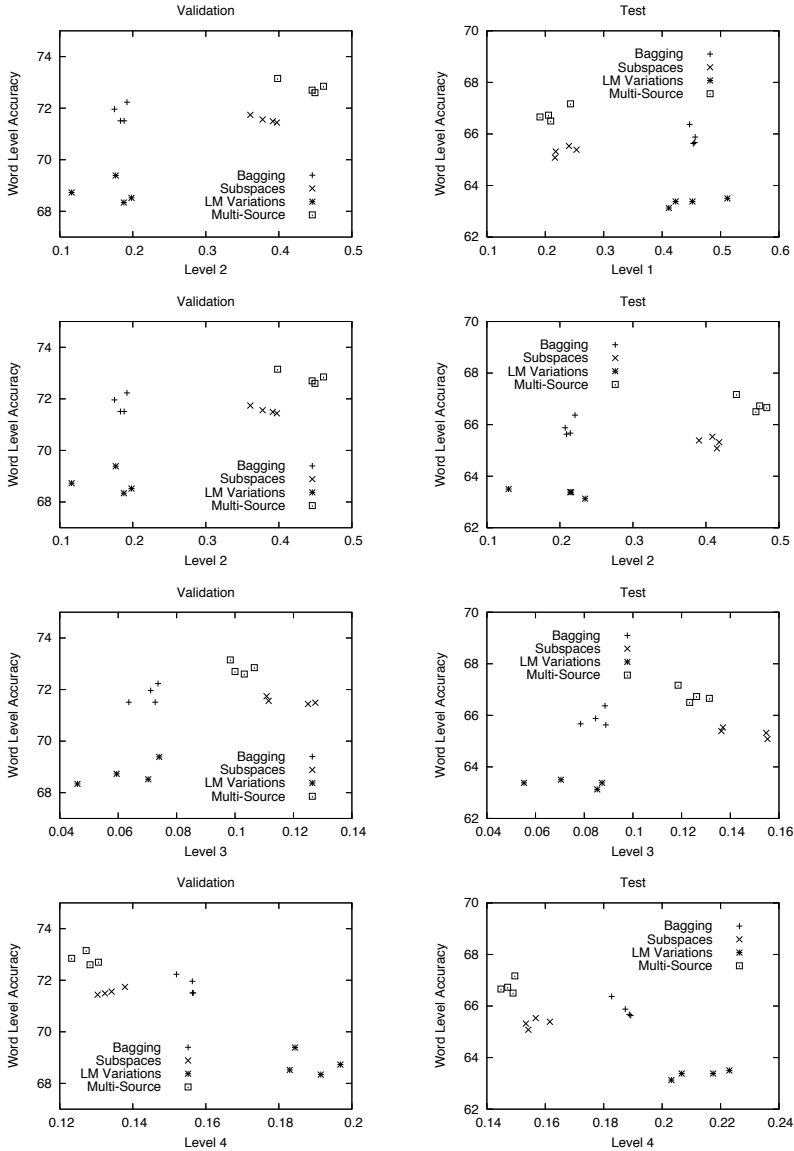
Figures 11.5 and 11.6 show the results of the diversity measures and the accuracy on the same set, i.e. either on the validation or on the test set. In practice, however, it would be useful to have a diversity measure that enables some sort of performance prediction for unknown data. Therefore, the diversity measures calculated on the validation set are plotted against the accuracy of the ROVER combination measured on the test set in Fig. 11.7. The diversity measure that best predicts the performance on the test set is the double fault measure. Most other diversity measures give no reliable predictions.



**Figure 11.4** Evaluation of the diversity measures on synthetic data. The x-axis shows the values of the diversity measure, and the ensemble accuracy is displayed on the y-axis. The pairwise diversity measures are given in the upper part, whereas the non-pairwise measures are shown in the lower part.

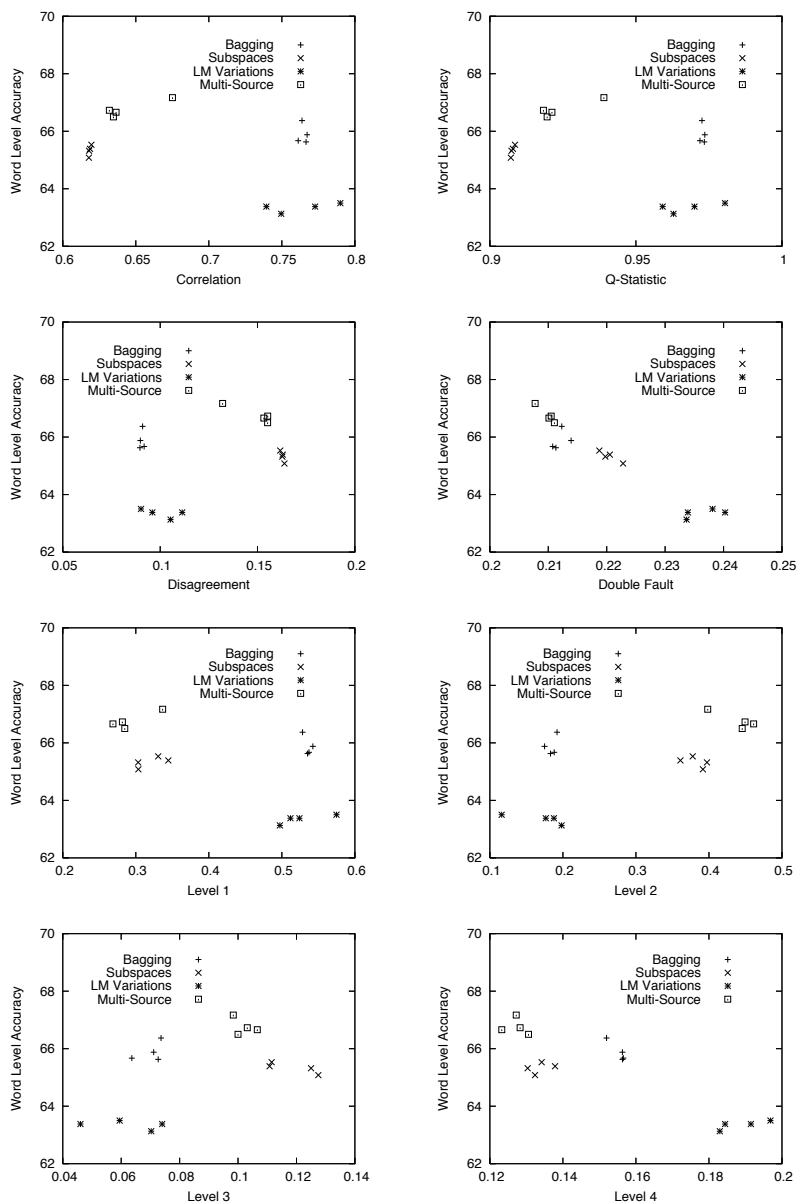


**Figure 11.5** Scatterplots of different pairwise diversity measures and ROVER combination accuracy. The plots on the left side are calculated on the validation set, and the test set results are shown on the right.



**Figure 11.6** Scatterplots of level-based non-pairwise diversity measures and ROVER combination accuracy. The plots on the left side are calculated on the validation set, and the test set results are shown on the right.





**Figure 11.7** Scatterplots of the prediction power of the applied diversity measures. The diversity measured on the validation set is plotted against the ROVER combination accuracy on the test set.

It is worth having a closer look at the results of the diversity analysis on the real-word ensembles reported in Figs. 11.5, 11.6, and 11.7 and compare them with the results on artificial data shown in Fig. 11.4. Whereas a clear correlation between the combination accuracy and most considered diversity measures can be reported for artificial data, only the double fault and the level 4 measure perform reasonably well on the real-world data. All other diversity measures estimate the diversity of the random feature subspace ensembles higher than the diversity of the Bagging ensemble, although combining the Bagging ensembles usually results in a higher recognition accuracy. This corresponds to the oracle performance of Bagging and feature subspace ensembles reported in Fig. 11.2. Thus, the conclusion can be drawn that the combination procedure is unable to fully benefit from the ability of the random feature subspace ensembles.

## 11.7 Discussion

This section concludes the experimental evaluation by discussing several interesting issues that were observed during the evaluation of the different ensemble methods.

In general, one can conclude that ensemble methods can successfully be applied to offline handwritten text line recognition. Most of the proposed methods increase the performance compared to the single reference system. Of course, the computational cost of the proposed ensemble methods is substantially higher than that of the single reference system raising the question whether the ensemble methods are worthwhile. This problem can be solved if multiple computational units are available. Then, the expensive calculations, i.e. the recognition conducted by the individual ensemble members, can be distributed and executed in parallel, and the required computation time is not substantially higher by applying the proposed ensemble methods.

In almost all experiments the Bagging method outperforms the corresponding feature subspace method, which itself performs better than the LM variation ensemble. Although LM variation is a fast and specific ensemble generation method, it did not produce good results in the conducted experiments. One reason for this fact may be that the combination methods, which are all based on an alignment, could not deal well with the different word lengths that result from the variation of the word insertion penalty. Additionally, the high double fault and the high level 4 diversity measure indicate that the diversity of these ensembles is rather low. The superior performance of the Bagging ensembles can be explained by the fact that during testing, a Bagging recogniser uses all available nine features, whereas only six out of nine features are used by a feature subspace recogniser.

Using multiple generation methods to build an ensemble, referred to as multi-source ensembles, consistently outperformed each corresponding single-source ensemble. The

diversity, the oracle performance, and the combination accuracy increase by considering multiple ensemble generation methods.

In many experiments a good confidence measure, i.e. Conf1 or Conf2, or a sophisticated combination method, e.g. LM-based decision, increased the performance. However, they often trade-off, i.e. not much improvement usually happens if a good confidence measure as well as a sophisticated combination method is used. This indicates that a good confidence measure enhances a rather weak combination strategy, and vice versa, a good combination strategy can compensate for the lack of a reliable confidence measure.

The proposed diversity analysis framework was used to describe the relationship between the ensemble performance and various diversity measures. However, the inclusion of diversity into the ensemble generation process is still an open issue. In combination with the ensemble performance on a validation set, ensemble diversity could be used during the ensemble member selection process.



## **Part III**

# **Conclusions and Outlook**



# 12

---

## Conclusions

This thesis investigates the use of ensemble methods for the recognition of general offline handwritten English text lines. The underlying recognition system is a state of the art recogniser based on hidden Markov models and statistical  $n$ -gram language modelling. Several ensemble creation methods and result combination schemes are discussed. An experimental evaluation on a large data set of handwritten text lines shows the effectiveness of the proposed methods.

After image normalisation, a sliding window method extracts a feature vector sequence from the image of a handwritten text. Each character is modelled with a hidden Markov model with an individual number of states. Based on a lexicon, word models are built. A bigram language model is used to build text line models. Three confidence measures, based on normalised likelihoods and alternative candidates, are calculated in a postprocessing step.

Three methods are applied to generate ensembles of recognisers from a single base recogniser: Bagging, which varies the training data, random feature subspace, that alters the feature vectors, and variation of the integration of the statistical language model, where the architecture is varied. An overproduce and choose method selects ensemble members using a greedy forward search. The most successful ensembles are created with the Bagging method. The ensembles further improve by using multiple ensemble generation methods to build multi-source ensembles.

The combination of multiple handwritten text line recognisers differs from most other multiple classifier combination tasks because the output of a text line recogniser is a sequence of words instead of a single class. The results of multiple text line recognisers are combined in two steps. First, an alignment method is needed to synchronise the recognition results which can be of different word-length. The result of the alignment is a word transition network. Second, a decision method is implemented that decides for each segment of the word transition network which word should be included in the

combination result. Various decision methods are investigated, including decision by frequency of occurrence, confidence-based voting, a statistical decision method, and a decision method that includes a statistical language model.

The experimental evaluation shows that the applied ensemble methods are able to increase the recognition performance compared to the optimised single recogniser. The most successful combination method is the decision based on language models. Additionally, the confidence measures based on alternative candidates are particularly able to improve the combination methods.



# 13

---

## Outlook

This chapter provides an incomplete list of issues that may be considered in future research to improve the state of the art in offline handwritten text line recognition.

In recent years, discriminative models, e.g. support vector machines, have become very popular in the pattern recognition community. Most of these models cannot be directly applied to handwritten text line recognition because sequences of classes must be recognised rather than just a single class. Recently, however, novel methods have been proposed that should be evaluated systematically on offline handwritten text data. The first method is called conditional random field and is a unidirectional discriminative model that can be used for the labelling of sequential data [78]. A conditional random field can be understood as a generalisation of an HMM that transforms constant transition probabilities into arbitrary functions. The second method is a special type of neural network called long short-term memory network and includes feedback connections as well as specific cells that can convey information over temporal distances [51].

The indexing of handwritten documents is an interesting application of the presented text recognition system. The task of indexing is different from the recognition task because not every word that occurs in a text is of interest for an index; e.g. words like *the*, *or*, and *a* occur in every English text and are typically neglected. Thus, the interesting words must be spotted and the recognition system should be optimised on these words.

Boosting and its numerous variations are widely considered to be very effective ensemble generation methods. Boosting is very expensive for the considered handwriting recognition task because the entire training set must be decoded in each iteration. Therefore, it has not been used in this thesis. Nevertheless, and with machines getting more powerful, applying Boosting to handwritten text line recognition may be an interesting topic for future research.

This thesis only considers ensembles that are automatically generated from a given base recogniser. The question remains open which combination methods are well suited if multiple base recognisers are used to generate the ensemble or if no automatic ensemble generation is considered, but all recognisers are built manually. In general, the diversity is increased with additional base recognisers. Thus, with an appropriate combination method the performance can increase.

# Bibliography

- [1] E. Anquetil and G. Lorette. Perceptual model of handwriting drawing application to the handwriting segmentation problem. In *Proc. 4th International Conference on Document Analysis and Recognition, Ulm, Germany*, pages 112–117, 1997.
- [2] R. Bakis. Continuous speech recognition via centisecond acoustic states. In *Proc. of the 91. Meeting of the Acoustic Society of America, Washington, USA*, 1976.
- [3] L. Bauer. *Manual of Information to Accompany the Wellington Corpus of Written New Zealand English*. Department of Linguistics, Victoria University, Wellington, New Zealand, 1993.
- [4] L. E. Baum and J. A. Egon. An inequality with application to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. *Bulletin of the American Mathematical Society*, 73:360–363, 1967.
- [5] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.
- [6] D. Black. *The Theory of Committees and Elections*. Cambridge University Press, 1963.
- [7] R. M. Bozinovic and S. N. Srihari. Off-line cursive script word recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(1):68–83, 1989.
- [8] A. Brakensiek and G. Rigoll. Handwritten address recognition using hidden Markov models. In A. Dengel, M. Junker, and A. Weisbecker, editors, *Reading and Learning*, pages 103–122. Springer, 2004.
- [9] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

- [10] E. Brill and R. C. Moore. An improved error model for noisy channel spelling correction. In *Proc. 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293, 2000.
- [11] A. Britto-Jr, R. Sabourin, E. Lethelier, F. Bortolozzi, and C. Suen. Improvement in handwritten numeral string recognition by slant normalization and contextual information. In *Proc. 7th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, The Netherlands*, pages 323–332, 2000.
- [12] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6:5–20, 2005.
- [13] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.
- [14] R. Bryll, R. Gutierrez-Osuna, and F. Quek. Attribute Bagging: improving accuracy of classifier ensembles by using random feature subsets. *Pattern Recognition*, 36(6):1291–1302, 2003.
- [15] H. Bunke. Recognition of cursive Roman handwriting - past, present and future. In *Proc. 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland*, pages 448–459, 2003.
- [16] L. Burnard. *Reference Guide for the British National Corpus (World Edition)*. British National Corpus Consortium by the Humanities Computing Unit at Oxford University Computing Services, 2000.
- [17] F. Camastra, M. Spinetti, and A. Vinciarelli. Cursive character challenge: a new database for machine learning and pattern recognition. In *Proc. 18th International Conference on Pattern Recognition, Hong Kong, China*, pages 913 – 916, 2006.
- [18] H. Cecotti and A. Belaïd. Hierarchical behavior knowledge space. In *7th International Workshop on Multiple Classifier Systems, Prague, Czech Republic*, LNCS 4472, pages 421–430. Springer, 2007.
- [19] M. Y. Chen, A. Kundu, and J. Zhou. Off-line handwritten word recognition using a hidden Markov model type stochastic network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):481–496, 1994.
- [20] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of Association for Computational Linguistics*, pages 310–318, 1996.

- [21] X. Chen, X. Liu, and Y. Jia. Learning handwritten digit recognition by the max-min posterior pseudo-probabilities method. In *Proc. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil*, pages 342–346, 2007.
- [22] M. Cheriet, N. Kharma, C.-L. Liu, and C. Suen. *Character Recognition Systems*. John Wiley & Sons Inc, 2007.
- [23] K. W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proc. 2nd Conference on Applied Natural Language Processing*, pages 136–143, 1988.
- [24] J. Daugman. Biometric decision landscapes. In *Technical Report No. TR482*. University of Cambridge Computer Laboratory, 2000.
- [25] C. O. de A. Freitas, L. S. Oliveira, E. Justino, and S. B. K. Aires. Zoning and metaclasses improving the character recognition. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, 2006.
- [26] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal Royal Stat. Soc., Series B*, 39(1):1–38, 1977.
- [27] T. Dietterich. Ensemble methods in machine learning. In *1st International Workshop on Multiple Classifier Systems, Cagliari, Italy*, pages 1–15, 2000.
- [28] R. P. Duin and D. M. Tax. Experiments with classifier combining rules. In *1st International Workshop on Multiple Classifier Systems, Cagliari, Italy*, pages 16–29, 2000.
- [29] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1999.
- [30] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, N.Y., 1993.
- [31] A. Elms, S. Procter, and J. Illingworth. The advantage of using an HMM-based approach for faxed word recognition. *International Journal on Document Analysis and Recognition*, 1(1):18–36, 1998.
- [32] G. Fink and T. Plotz. On appearance-based feature extraction methods for writer-independent handwritten text recognition. In *Proc. 8th International Conference on Document Analysis and Recognition, Seoul, Korea*, pages 1070–1074, 2005.

- [33] J. Fiscus. A post-processing system to yield reduced word error rates: recognizer output voting error reduction. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding, Santa Barbara*, pages 347–352, 1997.
- [34] G. D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.
- [35] W. N. Francis and H. Kucera. *Brown Corpus Manual. 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, USA, 1979.
- [36] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to Boosting. In *Proc. European Conference on Computational Learning Theory*, pages 23–37, 1995.
- [37] P. Gader, M. Mohamed, and J. Keller. Fusion of handwritten word classifiers. *Pattern Recognition Letters*, 17:577–584, 1996.
- [38] M. D. Garris. Design and collection of a handwriting sample image database. *Social Science Computing Journal*, 10:196–214, 1992.
- [39] B. Gatos, I. Pratikakis, A. Kesidis, and S. Perantonis. Efficient off-line cursive handwriting word recognition. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, pages 121–125, 2006.
- [40] B. Gatos, I. Pratikakis, and S. Perantonis. Hybrid off-line cursive handwriting word recognition. In *Proc. 18th International Conference on Pattern Recognition, Hong Kong, China*, pages 998–1002, 2006.
- [41] J. Ghosh. Multiclassifier systems: back to the future. In *3rd International Workshop on Multiple Classifier Systems, Cagliari, Italy*, LNCS 2364, pages 1–15. Springer, 2002.
- [42] G. Giacinto and F. Roli. Design of effective neural network ensembles for image classification purposes. *Image Vision and Computing*, 19(9-10):699–707, 2001.
- [43] I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3/4), 1953.
- [44] J. Goodman. A bit of progress in language modeling. Technical Report MSR-TR-2001-72, Microsoft Research, 2001.
- [45] N. Gorski. Optimizing error-reject trade off in recognition systems. In *Proc. 4th International Conference on Document Analysis and Recognition, Ulm, Germany*, volume 2, pages 1092–1096, 1997.

- [46] S. Günter and H. Bunke. Ensembles of classifiers for handwritten word recognition. *International Journal on Document Analysis and Recognition*, 5(4):224–232, 2003.
- [47] S. Günter and H. Bunke. HMM-based handwritten word recognition: on the optimization of the number of states, training iterations and Gaussian components. *Pattern Recognition*, 37:2069–2079, 2004.
- [48] T. K. Ho. The random space method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
- [49] T. K. Ho. Multiple classifier combination: lessons and next steps. In A. Kandel and H. Bunke, editors, *Hybrid Methods in Pattern Recognition*, pages 171–198. World Scientific, 2002.
- [50] T. K. Ho, J. J. Hull, and S. N. Srihari. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(1):66–75, 1994.
- [51] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
- [52] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, Massachusetts, 1979.
- [53] S. Hoque and M. Fairhurst. A novel scheme for implementation of the scanning n-tuple classifier in a constrained environment. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, pages 127–132, 2006.
- [54] Y. Huang and C. Suen. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):90–94, 1995.
- [55] J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.
- [56] N. Ide and C. Macleod. The American National Corpus: a standardized resource for American English. In *Proceedings of the Corpus Linguistics 2001 conference, Lancaster UK*, pages 274–281, 2001.
- [57] S. Impedovo, L. Ottaviano, and S. Occhiegro. Optical character recognition - a survey. *International Journal of Pattern Recognition and Artificial Intelligence*, 5:1–24, 1991.

- [58] A. Jain, R. Duin, and J. Mao. Statistical pattern recognition: a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.
- [59] F. Jelinek. Self-organized language modeling for speech recognition. *Readings in Speech Recognition*, pages 450–506, 1990.
- [60] S. Johansson, E. Atwell, R. Garside, and G. Leech. *The Tagged LOB Corpus, User's Manual*. Norwegian Computing Center for the Humanities, Bergen, Norway, 1986.
- [61] K. C. Jung and H. J. Kim. Korean character recognition using a TDNN and an HMM. *Pattern Recognition Letters*, 20(6):551–563, 1999.
- [62] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401, 1987.
- [63] E. Kavallieratou, N. Fakotakis, and G. Kokkinakis. An unconstrained handwriting recognition system. *International Journal on Document Analysis and Recognition*, 4(4):226–242, 2002.
- [64] E. Kavallieratou, K. Sgarbas, N. Fakotakis, and G. Kokkinakis. Handwritten word recognition based on structural characteristics and lexical support. In *Proc. 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland*, pages 562–566, 2003.
- [65] M. D. Kernighan, K. W. Church, and W. A. Gale. A spelling correction program based on a noisy channel model. In *Proc. 13th Conference on Computational Linguistics*, pages 205–210, 1990.
- [66] Y. Kessentini, T. Paquet, and A. Benhamadou. A multi-stream approach to off-line handwritten word recognition. In *Proc. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil*, pages 317–321, 2007.
- [67] G. Kim, V. Govindaraju, and S. Srihari. An architecture for handwritten text recognition systems. In S.-W. Lee, editor, *Advances in Handwriting Recognition*, pages 163–172. World Scientific Publ. Co., 1999.
- [68] J. Kittler, M. Hatef, P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226 – 239, 1998.
- [69] R. Kneser and H. Ney. Improved backing-off for m-gram language modeling. In *Proc. International Conference on Acoustics, Speech, and Signal Processing, Detroit, USA*, pages 181–184, 1995.



- [70] A. L. Koerich. Rejection strategies for handwritten word recognition. In *Proc. 9th International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan*, pages 479–484, 2004.
- [71] A. L. Koerich, A. S. Britto Jr., L. de Oliveira, and R. Sabourin. Fusing high- and low-level features for handwritten word recognition. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, 2006.
- [72] A. L. Koerich, R. Sabourin, and C. Y. Suen. Large vocabulary off-line handwriting recognition: a survey. *Pattern Analysis and Applications*, 6(2):97–121, 2003.
- [73] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. International Joint Conference on Artificial Intelligence*, pages 1137–1145, 1995.
- [74] U. Kressel and J. Schürmann. Pattern classification techniques based on function approximation. In H. Bunke and P. Wang, editors, *Handbook of Character Recognition and Document Image Analysis*, pages 49–78. World Scientific, 1997.
- [75] L. I. Kuncheva. A theoretical study on expert fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):281–286, 2000.
- [76] L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. John Wiley & Sons Inc, 2004.
- [77] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.
- [78] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289, 2001.
- [79] C.-L. Liu, H. Sako, and H. Fujisawa. Performance evaluation of pattern classifiers for handwritten character recognition. *International Journal on Document Analysis and Recognition*, 4(3):191–204, 2002.
- [80] J. Liu and P. Gader. Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. *Pattern Recognition*, 35(10):2061–2071, 2002.
- [81] M. Liwicki and H. Bunke. Combining on-line and off-line systems for handwriting recognition. In *Proc. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil*, pages 372–376, 2007.

- [82] M. Liwicki and H. Bunke. Handwriting recognition of whiteboard notes - studying the influence of training set size and type. *International Journal of Pattern Recognition and Artificial Intelligence*, 21(1):83–98, 2007.
- [83] G. Lorette. Handwriting recognition or reading? What is the situation at the dawn of the 3rd millenium? *International Journal on Document Analysis and Recognition*, 2(1):2–12, 1999.
- [84] L. M. Lorigo and V. Govindaraju. Offline arabic handwriting recognition: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):712–724, 2006.
- [85] S. Madhvanath and V. Govindaraju. Local reference lines for handwritten phrase recognition. *Pattern Recognition*, 32(12):2021–2028, 1999.
- [86] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer Professional Computing, New York, 2003.
- [87] L. Mangu, E. Brill, and A. Stolcke. Finding consensus among words: Lattice-based word error minimization. In *Proc. Eurospeech'99*, pages 495–498, 1999.
- [88] U.-V. Marti and H. Bunke. Use of positional information in sequence alignment for multiple classifier combination. In J. Kittler and F. Roli, editors, *2nd International Workshop on Multiple Classifier Systems, Cambridge, England*, LNCS 2096, pages 388 – 398. Springer, 2001.
- [89] U.-V. Marti and H. Bunke. Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system. *International Journal of Pattern Recognition and Artificial Intelligence*, 15:65–90, 2001.
- [90] U.-V. Marti and H. Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5:39 – 46, 2002.
- [91] S. Marukatat, T. Artieres, and P. Gallinari. Rejection measures for handwriting sentence recognition. In *Proc. 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-on-the-Lake, Canada*, pages 24–29, 2002.
- [92] J. Meynet. *Information theoretic combination of classifiers with application to face detection*. PhD thesis, École Polytechnique Fédéral de Lausanne, Switzerland, 2007.
- [93] M. Minsky. Logical versus analogical or symbolic versus connectionist or neat versus scruffy. *AI Magazine*, 12(2):34–51, 1991.

- [94] M. A. Mohamed and P. D. Gader. Handwritten word recognition using segmentation-free hidden markov modeling and segmentation-based dynamic programming techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):548–554, 1996.
- [95] A. Ogawa, K. Takeda, and F. Itakura. Balancing acoustic and linguistic probabilities. In *Proc. International Conference on Acoustics, Speech and Signal Processing, Seattle, USA*, volume 1, pages 181–184, 1998.
- [96] L. S. Oliveira, M. Morita, and R. Sabourin. Feature selection for ensembles applied to handwriting recognition. *International Journal on Document Analysis and Recognition*, 8(4):262 – 279, 2006.
- [97] Y.-H. Pao. Neural net computing for pattern recognition. In *Handbook of Pattern Recognition and Computer Vision*, pages 125–162. Eds. C. H. Chen, L. F. Pau and P. S. P. Wang, World Scientific Publishing Co. Pte. Ltd, Singapore, 1993.
- [98] D. Partridge and W. B. Yates. Engineering multiversion neural-net systems. *Neural Computation*, 8(4):869–893, 1996.
- [99] J. Pitrelli and M. P. Perrone. Confidence modeling for verification post-processing for handwriting recognition. In *Proc. 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-on-the-Lake, Canada*, pages 30–35, 2002.
- [100] J. Pitrelli and M. P. Perrone. Confidence-scoring post-processing for off-line handwritten-character recognition verification. In *Proc. 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland*, volume 1, pages 278–282, 2003.
- [101] J. F. Pitrelli, J. Subrahmonia, and M. P. Perrone. Confidence modeling for handwriting recognition: algorithms and applications. *International Journal on Document Analysis and Recognition*, 8(1):35–46, 2006.
- [102] R. Plamondon and S. Srihari. Online and off-line handwriting recognition: a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:63–84, 2000.
- [103] S. Quiniou and E. Anquetil. A priori and a posteriori integration and combination of language models in an on-line handwritten sentence recognition system. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, pages 403–408, 2006.
- [104] L. Rabiner. A tutorial on hidden Markov models and selected application in speech recognition. *Proc. of the IEEE*, 77(2):257–286, 1989.

- [105] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, 1993.
- [106] A. Rahman and M. Fairhurst. Multiple classifier decision combination strategies for character recognition: A review. *International Journal on Document Analysis and Recognition*, 5(4):166–194, 2003.
- [107] A. Rahmann and M. Fairhurst. Multiple expert classification: A new methodology for parallel decision fusion. *International Journal on Document Analysis and Recognition*, 3(1):40–55, 2000.
- [108] S. Raudys. Trainable fusion rules. i. large sample size case. *Neural Networks*, 19(10):1506–1516, 2006.
- [109] S. Raudys and F. Roli. The behavior knowledge space fusion method: analysis of generalization error and strategies for performance improvement. In *4th International Workshop on Multiple Classifier Systems, Guildford, UK*, pages 55–64, 2003.
- [110] J. R. Rico-Juan and J. M. Iñesta. Edit distance for ordered vector sets: a case of study. In *Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition, Hong Kong, China*, LNCS 4109, pages 200–207. Springer, 2006.
- [111] R. Rojas. *Neural Networks - A Systematic Introduction*. Springer-Verlag, Berlin, New-York, 1996.
- [112] F. Roli, G. Giacinto, and G. Vernazza. Methods for designing multiple classifier systems. In *2nd International Workshop on Multiple Classifier Systems, Cambridge, England*, LNCS 2096, pages 78–87. Springer, 2001.
- [113] R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? *Proc. of the IEEE*, 88:1270–1278, 2000.
- [114] S. Saitou, H. Yamada, and S. Mori. An analysis of hand-printed character data base III. *Bulletin of the Electrotechnical Laboratory*, 42(5):385–434, 1978.
- [115] A. Sanchis, V. Jimenez, and E. Vidal. Efficient use of the grammar scale factor to classify incorrect words in speech recognition verification. In *Proc. International Conference on Pattern Recognition, Barcelona, Spain*, volume 3, pages 278–281, 2000.
- [116] K. M. Sayre. Machine recognition of handwritten words: a project report. *Pattern Recognition*, 5(3):213–228, 1973.

- [117] H. Schwenk and J.-L. Gauvain. Improved ROVER using language model information. In *ISCA ITRW Workshop Automatic Speech Recognition: Challenges for the new Millenium, Paris*, pages 47 – 52, 2000.
- [118] A. Senior and A. Robinson. An off-line cursive handwriting recognition system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):309–321, 1998.
- [119] A. Sharkey and N. Sharkey. Combining diverse neural networks. *The Knowledge Engineering Review*, 12(3):231–247, 1997.
- [120] K. Sirlantzkis, M. Fairhurst, and M. Hoque. Genetic algorithms for multi-classifier system configuration: a case study in character recognition. In *2nd International Workshop on Multiple Classifier Systems, Cambridge, England, LNCS 2096*, pages 99–108. Springer, 2001.
- [121] D. Skalak. The sources of increased accuracy for two proposed boosting algorithms. In *Workshop on Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms*, 1996.
- [122] S. Smith, M. Bourgoin, K. Sims, and H. Voorhees. Handwritten character classification using nearest neighbor in large databases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):915–919, 1994.
- [123] N. Srihari, X. Yang, and G. R. Ball. Offline chinese handwriting recognition: an assessment of current technology. *Frontiers of Computer Science in China*, 1(2):137–155, 2007.
- [124] R. K. Srihari and C. M. Baltus. Incorporating syntactic constraints in recognizing handwritten sentences. In *Proc. International Joint Conference on Artificial Intelligence*, pages 1262–1267, 1993.
- [125] T. Steinherz, E. Rivlin, and N. Intrator. Offline cursive script word recognition – a survey. *International Journal on Document Analysis and Recognition*, 2(2-3):90–110, 1999.
- [126] C. Suen, C. Nadal, R. Legault, T. Mai, and L. Lam. Computer recognition of unconstrained handwritten numerals. *Proc. of the IEEE*, 80(7):1162–1180, 1992.
- [127] E. Taira, S. Uchida, and H. Sakoe. Nonuniform slant correction for handwritten word recognition. *IEICE Transactions on Information & Systems*, E87-D(5):1247–1253, 2004.

- [128] A. H. Toselli, V. Romero, E. Vidal, and L. Rodriguez. Computer assisted transcription of handwritten text images. In *Proc. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil*, pages 944–948, 2007.
- [129] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection Science*, 8(3-4):385–403, 1996.
- [130] T. Varga. *Off-line Cursive Handwriting Recognition Using Synthetic Training Data*. PhD thesis, Universität Bern, Switzerland, 2006.
- [131] T. Varga and H. Bunke. Comparing natural and synthetic training data for off-line cursive handwriting recognition. In *Proc. 9th International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan*, pages 221–225, 2004.
- [132] T. Varga and H. Bunke. Off-line handwritten word recognition using synthetic training data produced by means of a geometrical distortion model. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(7):1285–1302, 2004.
- [133] C. Viard-Gaudin, P. M. Lallican, P. Binter, and S. Knerr. The IRESTE on/off (IRONOFF) dual handwriting database. In *Proc. 5th International Conference on Document Analysis and Recognition, Bangalore, India*, page 455, 1999.
- [134] A. Vinciarelli. A survey on off-line cursive word recognition. *Pattern Recognition*, 35(7):1433–1446, 2002.
- [135] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of large vocabulary cursive handwritten text. In *Proc. 7th International Conference on Document Analysis and Recognition, Edinburgh, Scotland*, pages 1101 – 1105, 2003.
- [136] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):709–720, 2004.
- [137] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.
- [138] R. Wagner and M. Fischer. The string-to-string correction problem. *Journal of the ACM*, 21(1):168–173, 1974.
- [139] L. Wang and T. Jiang. On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4):337–348, 1994.

- [140] W. Wang, A. Brakensiek, and G. Rigoll. Combination of multiple classifiers for handwritten word recognition. In *Proc. 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-on-the-Lake, Canada*, pages 117–122, 2002.
- [141] T. Winderatt. Diversity measures for multiple classifier system analysis and design. *Information Fusion*, 6(1):21–36, 2004.
- [142] K. Woods, W. P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):405–410, 1997.
- [143] L. Xu, A. Krzyzak, and C. Y. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3):418–435, 1992.
- [144] X. Ye, M. Cheriet, and C. Y. Suen. StrCombo: combination of string recognizers. *Pattern Recognition Letters*, 23:381–394, 2002.
- [145] S. J. Young, N. H. Russell, and J. H. S. Thornton. Token passing: a simple conceptual model for connected speech recognition systems. CUED technical report F INFENG/TR38, Cambridge University, 1989.
- [146] U. G. Yule. On the association of attributes in statistics. *Royal Society of London Philosophical Transactions Series A*, 194:257–319, 1900.
- [147] T. Zeppenfeld, M. Finke, K. Ries, M. Westphal, and A. Waibel. Recognition of conversational telephone speech using the JANUS speech engine. In *Proc. International Conference on Acoustics, Speech, and Signal Processing, Munich, Germany*, pages 1815–1818, Munich, Germany, 1997.
- [148] M. Zimmermann, R. Bertolami, and H. Bunke. Rejection strategies for offline handwritten sentence recognition. In *Proc. 17th International Conference on Pattern Recognition, Cambridge, England*, volume 2, pages 550–553, 2004.
- [149] M. Zimmermann and H. Bunke. Automatic segmentation of the IAM off-line database for handwritten English text. In *Proc. 16th International Conference on Pattern Recognition, Quebec, Canada*, volume 4, pages 35–39, 2002.
- [150] M. Zimmermann and H. Bunke. Hidden Markov model length optimization for handwriting recognition systems. In *Proc. 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-on-the-Lake, Canada*, pages 369–374, 2002.

- [151] M. Zimmermann and H. Bunke. N-gram language models for offline handwritten text recognition. In *Proc. 9th International Workshop on Frontiers in Handwriting Recognition, Tokyo, Japan*, pages 203 – 208, 2004.
- [152] M. Zimmermann and H. Bunke. Optimizing the integration of a statistical language model in HMM based offline handwriting text recognition. In *Proc. 17th International Conference on Pattern Recognition, Cambridge, England*, volume 2, pages 541 – 544, 2004.
- [153] M. Zimmermann, J.-C. Chappelier, and H. Bunke. Offline grammar-based recognition of handwritten sentences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(5):818–821, 2006.



# Curriculum Vitae

## Personal Details:

Name: Roman Bertolami  
Address: Brückfeldstrasse 19  
CH-3012 Bern, Switzerland  
Date of Birth: 25th September 1978  
Hometown: Thun, Switzerland  
Nationality: Swiss  
Marital Status: Single

## Education:

2004 - 2008 PhD Candidate at the University of Bern, Switzerland  
1999 - 2004 Master of Science, Study of Computer Science, University of Bern, Minor Fields in Mathematics and General Ecology  
1994 - 1998 Matura Typus B, Grammar School, Thun  
1985 - 1994 Primary & Secondary School, Liestal and Thun

## Employment:

2004 - present Research Assistant in Computer Vision and Artificial Intelligence, University of Bern, Switzerland  
2003 Specialist at the Computer Services Departement of the University of Bern, Switzerland  
2002 Lecture Assistant in Practical Course on Software Engineering, Software Composition Group, University of Bern, Switzerland  
1999 - 2003 Employee at the Computer Services Departement of the Federal Office of Transportation, Bern, Switzerland



# Conference Contributions

## Oral Presentations

- R. Bertolami. *Rejection Strategies for Offline Handwritten Sentence Recognition*. 17th International Conference on Pattern Recognition, Cambridge, United Kingdom, August 2004.
- R. Bertolami. *Multiple Handwritten Text Recognition Systems Derived from Specific Integration of a Language Model*. 8th International Conference on Document Analysis and Recognition, Seoul, Korea, August 2005.
- R. Bertolami. *Non-Uniform Slant Correction for Handwritten Text Line Recognition*. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, September 2007.

## Poster Presentations

- R. Bertolami. *Early Feature Stream Integration Versus Decision Level Combination in a Multiple Classifier System for Text Line Recognition*. 18th International Conference on Pattern Recognition, Hong Kong, China, August 2006.
- R. Bertolami. *Diversity Analysis for Ensembles of Word Sequence Recognisers*. Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition, Hong Kong, China, August 2006.
- R. Bertolami. *Combination of Multiple Handwritten Text Line Recognition Systems with a Recursive Approach*. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France, October 2006.
- R. Bertolami. *Multiple Classifier Methods for Offline Handwritten Text Line Recognition*. 7th International Workshop on Multiple Classifier Systems, Prague, Czech Republic, May 2007.



# List of Publications

## Journal Articles

- R. Bertolami, M. Zimmermann, and H. Bunke. Rejection strategies for offline handwritten text line recognition. *Pattern Recognition Letters*, 27(16), pages 2005-2012, 2006.
- R. Bertolami and H. Bunke. Integration of n-gram language models in multiple classifier systems for offline handwritten text line recognition. *accepted for publication at International Journal of Pattern Recognition and Artificial Intelligence*.
- R. Bertolami and H. Bunke. Hidden Markov model based ensemble methods for offline handwritten text line recognition. *accepted for publication at Pattern Recognition*.
- A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A Novel Connectionist System for Improved Unconstrained Handwriting Recognition. *submitted*.
- R. Bertolami, S. Uchida, M. Zimmermann, and H. Bunke. Recognition of Non-Uniformly Slant Corrected Offline Handwritten Text Lines. *submitted*.

## Conference Papers

- M. Zimmermann, R. Bertolami, and H. Bunke. Rejection strategies for offline handwritten sentence recognition. In *Proc. 17th International Conference on Pattern Recognition, Cambridge, England*, volume 2, pages 550-553, 2004.
- R. Bertolami and H. Bunke. Multiple handwritten text recognition systems derived from specific integration of a language model. In *Proc. 8th International Conference on Document Analysis and Recognition, Seoul, Korea*, volume 1, pages 521-524, 2005.
- R. Bertolami and H. Bunke. Ensemble methods for handwritten text line recognition systems. In *Proc. International Conference on Systems, Man and Cybernetics, Hawaii, USA*, pages 2334-2339, 2005.

- R. Bertolami and H. Bunke. Diversity analysis for ensembles of word sequence recognisers. In *Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition, Hong Kong, China*, LNCS 4109, pages 667-686. Springer, 2006.
- R. Bertolami and H. Bunke. Early feature stream integration versus decision level combination in a multiple classifier system for text line recognition. In *Proc. 18th International Conference on Pattern Recognition, Hong Kong, China*, volume Vol. II, pages 845-848, 2006.
- R. Bertolami, B. Halter, and H. Bunke. Combination of multiple handwritten text line recognition systems with a recursive approach. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition, La Baule, France*, pages 61-65, 2006.
- R. Bertolami and H. Bunke. Multiple classifier methods for offline handwritten text line recognition. In *7th International Workshop on Multiple Classifier Systems, Prague, Czech Republic*, LNCS 4472, pages 72-81. Springer, 2007.
- R. Bertolami, S. Uchida, M. Zimmermann, and H. Bunke. Non-uniform slant correction for handwritten text line recognition. In *Proc. 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil*, pages 18-22, 2007.
- R. Bertolami and H. Bunke. Ensemble methods to improve the performance of an english handwritten text line recognizer. In *Arabic and Chinese Handwriting Recognition: Summit, SACH 2006, College Park, USA, September 27-28, 2006, Selected Papers*, LNCS 4768, pages 265-277, Springer, 2008
- R. Bertolami and H. Bunke. Including language model information in the combination of handwritten text line recognisers. *submitted*.
- R. Bertolami, C. Gutmann, H. Bunke, and A. L. Spitz Shape Code Based Lexicon Reduction for Offline Handwritten Word Recognition. *submitted*.



