Technical Report
CVL-TR-10

Automatic Recognition of Weather Records

Manuel Keglevic and Robert Sablatnig

Computer Vision Lab
Institute of Computer Aided Automation
Vienna University of Technology

July 11, 2013

Abstract

Even before the use of computers, meteorological data was recorded for statistics by means of handwritten notes. In addition to those handwritten documents stored in historic archives, processes, as manually filling in measurement values into forms, still depend on pen and paper. However, manually digitizing this data for further processing is cumbersome. In this report, a character recognition system for automatically digitizing handwritten weather records is proposed. The weather record dataset regarded in this report consists of known printed forms with handwritten meteorological measurements. The scope of this report are the numeric temperature values which are measured at three different points of time.

The localization of the numerical data is achieved by first reconstructing the tabular structure of the form. Using vertical and horizontal projection profiles, the rough positions of the lines building up the table are found. Errors in the layout analysis are corrected using a-propri information of the form. Additionally, to reduce the influence of spurious lines, a stroke preserving line removal method is proposed. The extraction of the digits and signs is done using a binarization based on the Savakis filter. Using a trained PCA basis as a filterbank the features of the digits and signs are extracted and subsequently classified with multiple SVM with RBF kernels. Moreover, to allow an identification of uncertain prediction results, the class probabilities are estimated.

The evaluation was conducted using three different digit databases with manually annotated ground truth, synthetically generated digit images with spurious lines and weather records from five different measurement stations. On a dataset with weather records from five different measurement stations an accuracy of 93% per digit is achieved. Furthermore, on a dataset containing only weather records from a single writer the performance is improved to over 99%.

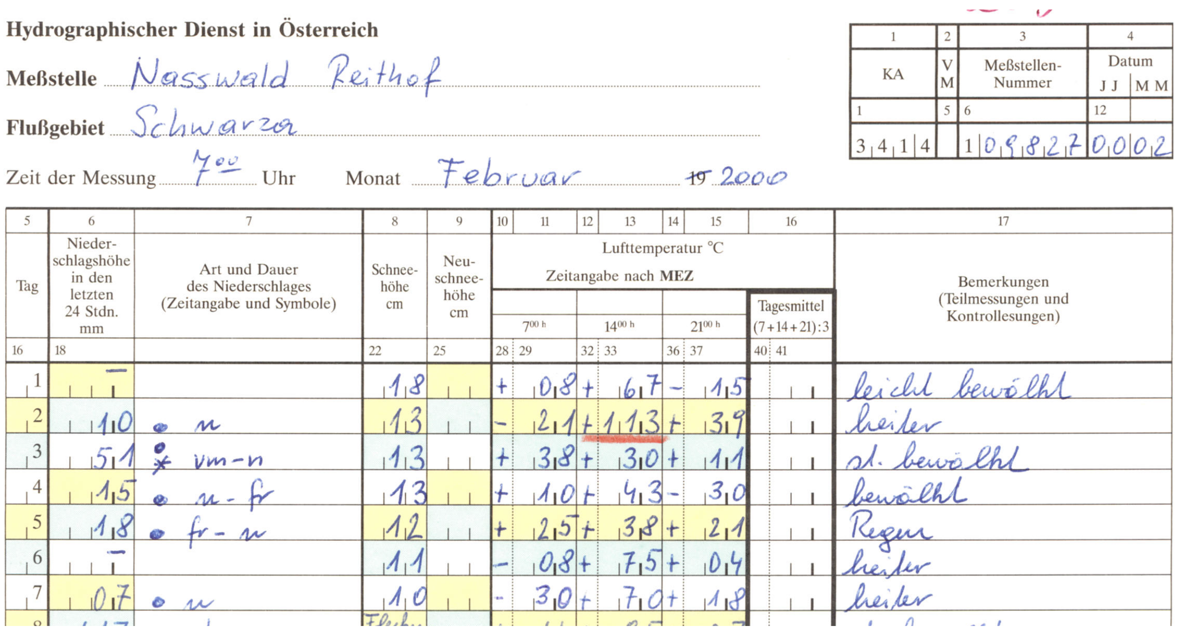
Contents

Chapter 1

Introduction

Even before the development of the first instruments which allowed precise meteorological measurements by Galileo Galilei at the end of the 16th century, meteorological observations have been documented [Man74, BDL$^+$10]. Further, information as the start of harvest, reports on weather damage, the opening of northern harbors, etc. can be used as proxy data [BDL$^+$10]. Using these meteorological recordings allows the study of historic climate with the aim to understand current and future climate as this allows for instance the investigation of fluctuations, i.e. climate variability vs. climate change [ZC89]. The longest continuous instrumental measurement series range back to the end of the 17th century [BKH$^+$06, BJH$^+$10], but not until the end of the 18th century the individual instrumental measurements were combined in national and international networks [Pfi01]. Even though organizations as for instance the Nation Climatic Data Center (NCDC) started to store the measurements digitally in the 1940s, there are still handwritten records stored in historic archives. In addition, processes, as manually filling in measurement values into forms, still depend on pen and paper [Hei88]. However, accessing the information stored in those documents requires time and manpower [RVF12]. By digitizing those documents, the advantages of digitally stored information, such as ease of access and statistical processing, can be exploited for handwritten documents [RVF12].

The documents discussed in this report are weather records which consist of known printed forms with handwritten meteorological measurements. The origin of these recordings are measurement stations in Lower Austria. The forms are filled out by the person responsible for the measurement station and subsequently aggregated by the Department of Hydrology and Geoinformation of the federal state government of Lower Austria (Amt der NiederÃ¶sterreichischen Landesregierung - Abteilung Hydrologie und Geoinformation). Each record includes daily measurement values of several meteorological parameters, namely: height of precipitation, type and duration of the precipitation, depth of snow, depth of fresh snow, temperature values and additional remarks. The scope of this report are the numeric temperature values which are measured at three different points of time. In Figure 1.1 an image detail of an exemplary weather record is shown with the temperature values in the central column.


Hydrographischer Dienst in Österreich

Meßstelle Nasswald Reithof

Flußgebiet Schwarza

Zeit der Messung 7^{00} Uhr Monat Februar ~~19~~ 2000

1	2	3	4	
KA	VM	Meßstellen-Nummer	Datum JJ	MM
1	5	6	12	
3414		109827	00	02

5	6	7	8	9	10	11	12	13	14	15	16	17
Tag	Niederschlagshöhe in den letzten 24 Stdn. mm	Art und Dauer des Niederschlages (Zeitangabe und Symbole)	Schneehöhe cm	Neuschneehöhe cm	Lufttemperatur °C Zeitangabe nach **MEZ** $7^{00\,h}$		$14^{00\,h}$		$21^{00\,h}$		Tagesmittel (7+14+21):3	Bemerkungen (Teilmessungen und Kontrollesungen)
16	18		22	25	28	29	32	33	36	37	40 41	
1	–		18		+	0,8	+	6,7	–	1,5		leicht bewölkt
2	1,0	● n	13		–	2,1	+	11,3	+	3,9		heiter
3	5,1	● ✱ vm–n	13		+	3,8	+	3,0	+	1,1		st. bewölkt
4	1,5	● n–fr	13		+	1,0	+	4,3	–	3,0		bewölkt
5	1,8	● fr–n	12		+	2,5	+	3,8	+	2,1		Regen
6	–		11		–	0,8	+	7,5	+	0,4		heiter
7	0,7	● n	10		–	3,0	+	7,0	+	1,8		heiter

Figure 1.1: Weather record from February of the year 2000 from the measurement station Nasswald in Lower Austria. The temperature values are located in the column "Lufttemperatur °C".

1.1 Motivation

Although most measurement stations in Austria are digital, manual observations are still used as a reference to detect incorrect measurements of the automatic stations[1]. The main reason that handwritten forms have not been replaced by electronic means is, that since the measuring stations in scope of this report are maintained voluntarily, the current work-flow is designed to be as convenient as possible for the maintainers. Additionally, the use of pen and paper requires no training and instructions. Simultaneously, this is advantageous in case the maintainer is indisposed and someone else has to temporarily assume the responsibility of recording the measurement values. Furthermore, the forms allow a certain degree of freedom as additional notes can be taken and observations can be recorded using meteorological symbols as shown in the third column in Figure 1.1. Besides all of this, the work-flow is approved and the maintainers are accustomed to it.

However, this requires that the handwritten forms containing the measurements are sent to the Department of Hydrology and Geoinformation of the federal state government of Lower Austria where they are manually transcribed and stored in a database. The aim of this report is to develop a system for the automatic transcription of those weather records. This way, the manpower needed for the cumbersome transcription is decreased, whereas the convenience of the work-flow for the maintainers of the measurement stations is retained. Moreover, the archives of the Department of Hydrology and Geoinformation still contain weather records which have not yet been digitized. By allowing for an automatic transcription of those records, valuable historic meteorological information can be

[1]http://www.zamg.ac.at/cms/de/klima/messnetze/wetterstationen

utilized with minimal human effort too.

1.1.1 Scope of Discussion

The focus of this report is the recognition of digits and signs in the previously described weather records. The scientific question can be formulated as: *Can the transcription of handwritten numerical data in weather records be automated using state-of-the-art digit recognition?*

As the tabular structure of the weather records can be defined beforehand, it is not intended to perform an exhaustive layout analysis but instead to utilize the a-priori information about the forms. This also means, that the localization of the handwritten numerical values is solely based on the reconstructed tabular structure and therefore the method proposed in not applicable for the general detection of handwriting in images.

Further, the individual digits of the temperature values in the records are separated into different cells. Therefore, the handwriting recognition is limited to segmented individual digits instead of whole digit strings. Additionally, the recognition of handwritten words, symbols or printed text is not regarded.

Finally, since the topic of this report is computer vision, the predicted digits and signs are not assessed using empirical temperature values for the region of measurement or other meteorological models.

1.1.2 Objective

Even though the first Optical Character Recognition (OCR) systems for machine printed text emerged in the 1950s, the recognition of handwriting is still an open topic. Moreover, although the digit recognition results on handwritten digit databases, as for instance MNIST, which is explained in detail in Section 4.1.1, are promising[2], it has to be considered that these digits have been segmented manually and therefore the results are not representative for real world data.

Hence, the weather records regarded provide several challenges: firstly, due to the scanning process the images may not be precisely upright which complicates the layout analysis and additionally, this influences the recognition of handwritten digits, as their features are not orientation invariant. For example the digits '6' and '9' cannot be distinguished without their orientation. Thus the rotation has to be corrected. Furthermore, for the recognition of the digits and signs they have to be located beforehand using the tabular structure of the form. However, since the appearance of the forms changed over the years, forms with similar structures but different appearance have to be handled. Moreover, although the structure of the forms suggests areas which are applicable for handwriting, writers are not bound to these constraints. This leads to digits and sign strokes crossing the form lines which have to be addressed since spurious lines in handwritten digit images impair the recognition accuracy as shown in Section 4.2.

Furthermore, as each measuring station is maintained by a different person, there is a great variation in writing styles. This includes for instance differences in size, slant, etc. and even in the structure of the digits. As an example, the digit '7' may be written

[2]http://yann.lecun.com/exdb/mnist/

with or without a central horizontal stroke. A contributing factor for this is the writer's origin. However, even the handwriting of a single writer exhibits variations. In Figure 1.2 variations of the digits '1', '8' and '9' are shown. Hence, the digit recognition must be able to deal with high intraclass variability.

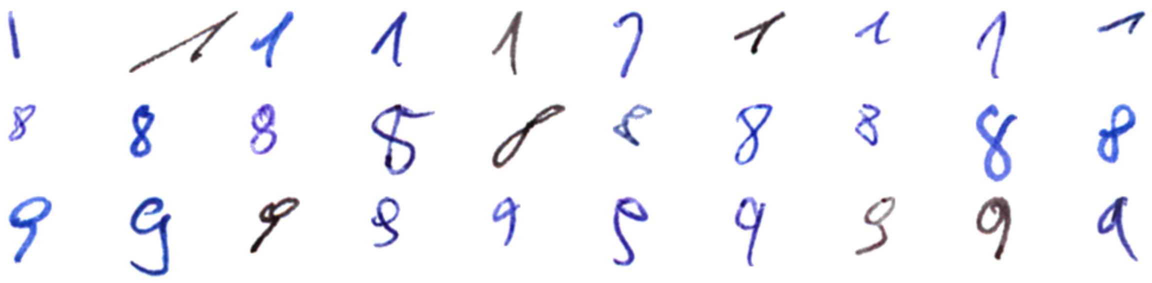


Figure 1.2: Variation in writing styles for the digits '1', '8' and '9'. These images were extracted from the CVL database which is described in more detail in Section 4.1.1

.

1.1.3 Main Contribution

The main contribution of this report is an assessment of the performance and the challenges of state-of-the-art digit recognition in a real world scenario. Therefore, in this report digit recognition is combined with document analysis techniques for the specific task of recognizing numerical temperature values in weather forms.

The digit recognition performance is evaluated using three different databases. In addition to overall performance values the challenges for different digit classes are examined in detail. However, as the system is intended to automate an otherwise manual process, it is important to quantify the human error to allow a comparison. Therefore, an experiment is conducted assessing the performance of human participants annotating handwritten digits.

As previously mentioned, parts of the frame lines may overlap with digits and signs, therefore a gradient-based approach for the stroke preserving line removal is proposed. In order to assess the performance of this approach, first the influence of spurious lines on the recognition accuracy is investigated. This allows on the one hand the evaluation of the performance gain of the proposed method and on the other hand reveals its limitations.

Finally, the performance and challenges of recognizing handwritten digits in the weather records is evaluated and compared to the results achieved on digit databases.

1.2 Definition of Terms

In this section, commonly used terms will be described and a list of abbreviations will be given.

CNN	**Convolutional Neural Networks:** neural network for the recognition of handwritten digits which incorporate the feature extraction process into the classifier using convolution and subsampling layers [LBBH98].
k-NN	**k-Nearest Neighbor:** simple classifier based on the Bayes decision theory which is often used as a baseline classifier [MNY99].
IDM	**Image Distortion Model:** model for matching the pixels of two images which only considers absolute positions. It additionally defines a distance function between the images [KDGN07].
OCR	**Optical Character Recognition:** subtopic of pattern recognition which deals with the recognition of characters in printed documents [MNY99].
PCA	**Principal Component Analysis:** a statistical method for finding an orthonormal basis that transforms the data to a new coordinate system in which the variance of the projected data in each D-dimensional subspace is maximal [Bis06].
NN	**Neural Networks:** classifiers built with the intention to replicate the human brain. NN consist of interconnected neurons. The inputs of a neuron are weighted and summed up in a summing unit and the output is subsequently computed using an activation function [MNY99].
RBF	**Radial Basis Function:** function which is solely dependent on the distance to a center. It is commonly modeled using a Gaussian and it can be used as kernel function for SVMs [Bis06].
ROI	**Region of Interest:** area of an image which contains the relevant information.
SVM	**Support Vector Machine:** binary classifier which separates the classes by fitting a hyperplane with maximal margin. Using a kernel function the sample vectors can be mapped into a higher dimensional feature space [Vap99].

1.3 Results

The system proposed was evaluated using three different digit databases with manually annotated ground truth, synthetically generated data for the evaluation of the line removal approach and weather records from five different measuring stations.

The accuracy on the handwritten digit databases is between 93.27% and 99.24%. The highest accuracy is achieved on the MNIST database whereas the lowest result is achieved on the CVL database for it is the most challenging since it is closest to real world scenarios.

The digits ‘8’ and ‘9’ were identified as the most problematic digit classes. Furthermore, an experiment was conducted to quantify the human error for annotating handwritten digits. The overlap of two participants has shown to be between 83.76% and 98.42% depending on the digit. Additionally, the prediction values computed proved to provide meaningful results as the correlation to the classification accuracy is explained by a monotonically increasing function.

Additionally, the devastating influence of spurious horizontal and vertical lines to the recognition accuracy was identified. Depending on the position and intensity of the lines a recognition accuracy of 99% drops to as low as 23%. Using the stroke preserving line removal approach proposed, the drop is reduced to 10-15% in the worst case and for most line intensities the maximal recognition performance is retained.

Finally, for the weather records an accuracy of 99.54% per digit was achieved on a dataset created by a single writer. However, for a second dataset containing at least five different writers the recognition accuracy drops to 87.9%. By increasing the number of training samples the performance was again increased to 93.98% and the remaining difference was accounted to different writing styles and the segmentation process.

1.4 Structure

This report is divided as follows: in the next chapter, related work will be presented. This includes state-of-the-art methods for the recognition of handwritten digits and approaches for the analysis of forms. Additionally, systems will be presented which focus on the recognition of handwritten digits in documents.

Subsequently, in Chapter 3, the parts of the system will be described in detail. Starting from scanned weather records, the methods used for analyzing the tabular structure, removing spurious lines, extracting the digits and signs, computing the features and for training and prediction of the supervised learning algorithm are presented and discussed.

In Chapter 4, firstly the handwritten digit recognition approach is evaluated. This is carried out using three different databases. Further, the human error is quantified by investigating the results of a manual annotation process. Synthetically generated digit images containing spurious lines are subsequently used to asses their influence and evaluate the performance of the proposed line removal algorithm. Finally, weather records from five different measurement stations are used to investigate the strengths, weaknesses and limitations of the entire system.

Finally, in the last chapter, the advantages and disadvantages of the proposed system are discussed and suggestions for improvements are depicted as future work.

Chapter 2

Related Work

For the automatic recognition of weather records methods from two topics, namely form recognition and handwritten digit recognition are combined. The purpose of this chapter is to give an overview of different techniques useful for the digit recognition in documents. However, it is not intended to provide an extensive survey but instead to present exemplary methods.

The chapter is divided as follows: first, a short overview of the state-of-the-art in handwritten digit recognition is given. This includes normalization, classification, and digit recognition systems. Second, document analysis techniques for documents containing pre-printed forms and approaches for digit recognition in documents in general are shown. In the end, a summary is provided.

2.1 Handwritten Digit Recognition

Handwritten digit recognition is a subtopic of Optical Character Recognition (OCR). The first OCR systems emerged in the 1950s [MNY99]. In the 1960s and 1970s improvements in hardware and research in this area widened the range of possible applications as for instance the automatic recognition of bank checks and postal addresses [MNY99]. However, OCR algorithms of that time used simple template matching [MNY99]. This restricted the usage to constrained applications with machine printed characters of a reduced alphabet, i.e. latin [AYV01]. Further, these OCR systems failed when challenged with poor print quality or unknown fonts [AYV01]. To overcome these problems, custom OCR fonts were introduced to improve recognition performance [MNY99].

Development of more powerful computers lead to more complex OCR systems which enabled the recognition of handwritten characters [MNY99]. The field can be separated into on-line and off-line character recognition depending on how the writing is digitized [PS00]. In case of on-line recognition the acquisition device outputs a series of (x, y) coordinates, i.e. the stroke is captured as a spatial coordinate over time:

$$f_1(t) = (x, y) \tag{2.1}$$

This method was for instance used in PDAs. Recently, whiteboards with special pens which report their position are used to obtain on-line handwriting data [GLF+09]. On the other hand, in off-line character recognition, the handwriting is created using pen and paper and digitized using an image scanner. The captured data is illumination information over spacial coordinates:

$$f_2(x, y) = I \tag{2.2}$$

where I is the intensity value of the pixel. This means that in case of on-line character recognition the stroke is directly available through the function f_1, i.e. for every time sample there is exactly one stroke position. In contrast, the function f_2 only contains illumination information. To retrieve the actual stroke it has to be separated from the background. Additionally, in contrast to on-line recognition the stroke has some unknown width and the temporal information is lost. However, there are application where on-line data is not accessible like the recognition of postal addresses [SGS93], filled out forms [GBC$^+$95], bank checks [PG08], etc.

There are several surveys and books which focus on handwriting recognition [LNSF03], [PS00], [AYV01] and [MNY99]. Due to the topic of this report, only off-line character recognition methods are presented here.

The OCR process can be divided into three stages [MNY99]:

1. In the *Preprocessing* stage, the handwritten characters are prepared for further processing. Depending on the algorithms used in succeeding stages this implies separating the handwritten stroke from the background, binarization of the image, contour calculation and size normalization.

2. In the *Feature Extraction* stage, certain traits of the images, i.e. features, are extracted to allow classification. This can be shape features, features describing the contour, properties of the gradient, responses to certain filters, the illumination information of the image itself, etc. Different writers have different handwriting styles, therefore it is desirable that the features extracted are invariant to variations in rotation, translation, scale, skew and local distortions [TJT96]. In case the features are not invariant to these distortions, the images have to be normalized.

3. In the last step, the *Classification*, a classifier is used to predict the class of the handwritten character. All classifiers described in this work are supervised and either use training data directly or learn it beforehand.

Even though in most digit recognition systems those stages are present, they are highly interwoven. This means that certain classifiers only work well on certain features and the other way round. Additionally, the extracted features rely on the normalization steps performed. Further, there are approaches which incorporate the feature extraction process into the classifier.

This section is divided into three subsections. First, an overview of different normalization techniques is given. Second, a selection of popular classifiers is described. Third, normalization, classification and feature extraction are brought together as state-of-the-art digit recognition systems are presented.

2.1.1 Normalization

The goal of character normalization is to reduce the intraclass variability to improve the recognition accuracy [MNY99]. The normalization process involves the mapping from an original image to a normalized image plane with fixed dimensions, i.e. the standard plane. The coordinate mapping functions define how the image is mapped to the standard plane. For instance using a linear mapping function the coordinates in the original image are simply multiplied by a coefficient. Moment based mapping is similar, however the central moment, i.e. the center of mass, is translated to the center of the standard plane [LNSF04]. This method is for instance used by Lecun et. al. [LBBH98] for the MNIST database and is shown by F2 in Figure 2.1. For nonlinear mapping, the accumulated line density histograms are used as coefficients (see F6 in Figure 2.1). An example for different normalization techniques is shown in Figure 2.1. The mappings F1 and F7 preserve the aspect ratio whereas the other mappings change it.

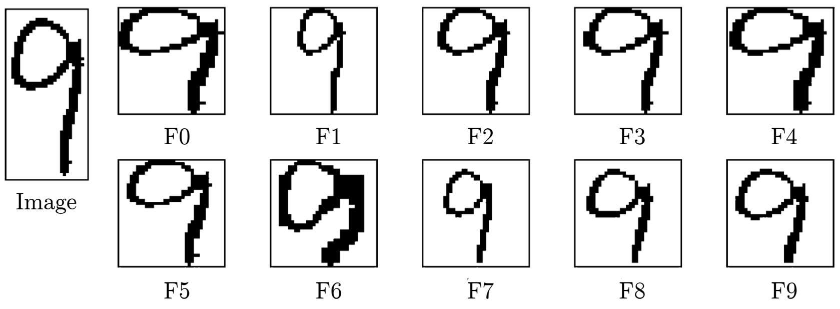


Figure 2.1: Comparison of different normalization techniques on a binary image (left) [LNSF04].

In addition to size normalization, deslanting can be used to correct the slant in writing styles to further reduce the intraclass variability. This can be done using second-order moments [LNSF04] or by fitting a regression line through the centroid of the image [TL02].

2.1.2 Classification

The purpose of the classifier is to assign each test sample a label representing its class or a membership score, i.e. the probability that the sample belongs to a specific class. In case of digit recognition the classes are the digits 0-9. All algorithms presented here learn a set of known pre-labeled samples, i.e. the training set, in order to predict unknown test samples.

In this section popular classifiers for digit recognition and character recognition in general are presented. This section is divided as follows: First, a baseline classifier, the k-Nearest Neighbor (k-NN) classifier is presented. Then, Neural Networks (NN) and Support Vector Machine (SVM) classifiers are shown.

k-Nearest Neighbor

The K-NN classifier is based on the Bayes decision theory, i.e. on a probability density estimation approach [MNY99]. Given a feature vector $\mathbf{x}$ of a test sample, the posterior probabilities $p(w_i \,|\, \mathbf{x})$, i.e. the probability that the sample belongs to class w_i considering the feature vector, is given by:

$$p(w_i \,|\, \mathbf{x}) = \frac{k_i}{k} \tag{2.3}$$

with k_i defining the number of training samples of class w_i falling into the window of the nearest k points [MNY99]. From this definition, the K-NN decision rule can be constructed:

1. Compute the k nearest neighbors to the test sample.
2. Assign the class label of having the most nearest neighbors.

The decision boundary of a K-NN classifier with $k = 1$ (the class of the nearest neighbor is assigned) can be visualized as a Voronoi diagram, if the Euclidean distance is used. The drawback of this method is that in order to classify one test sample the distances to all training samples have to be computed [MNY99]. This entails high computational effort and storage costs [MNY99]. Further, the parameter k has to be chosen carefully to cope with noise and over- and under-fitting. However, because of the fairly high classification performance this method is often used as a baseline classifier to evaluate other classifiers or investigate the quality of different features [MNY99].

Neural Networks

NN were invented with the intention of replicating the human brain in hopes of making intelligent machines [MNY99]. The basic building blocks of NN are interconnected neurons. The inputs of a neuron are weighted and summed up in a summing unit. After adding a bias, the output is computed using an activation function. In Figure 2.2 the model of a neuron is depicted. A commonly used activation function is the logistic function [MNY99]. It is continuous and derivable which is advantageous for learning and further, the output values can be used as posterior probabilities [MNY99].

The simplest NN are single-layer NN, which have only one layer of neurons. The input signals are connected to all neurons and the combined outputs of the neurons form the final output. However, single-layer NN provide only a linear decision boundary [MNY99]. To overcome this problem the Multi-Layer Perceptron (MLP) has one or more hidden layers [MNY99]. To explain the training of NN, it can be formulated as follows [LBBH98]:

$$Y^p = F(Z^p, W) \tag{2.4}$$

Starting form an input pattern Z^p the learning machine predicts an output Y^p using adjustable weights W and a function F. As input patterns Z^p for instance the pixel values of the input image or a precomputed feature vector can be used. In order to allow training, a loss function which measures how well the system can predict the class label is

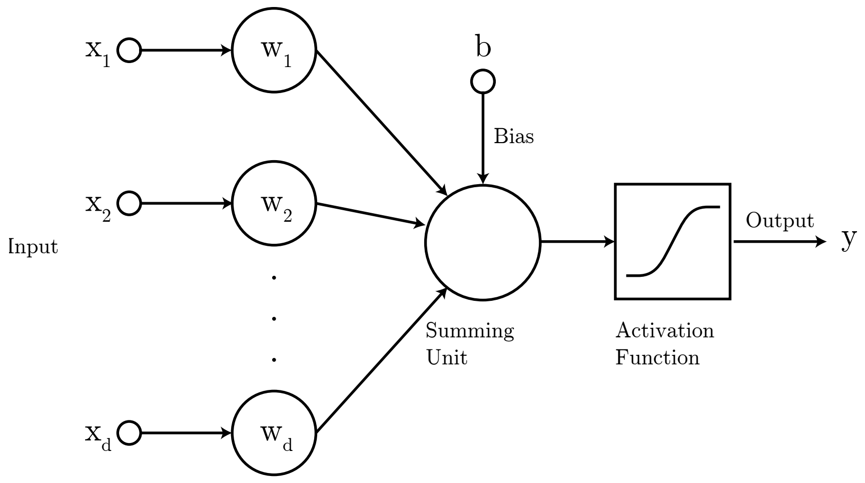


Figure 2.2: Model of a neuron. In the summing unit the weighted input is summed up and a bias is added. The activation function (in this case the logistic function) maps the result into a specified interval [MNY99].

defined as $E^p = D(D^p, Y^p)$. The distance function D measures the discrepancy between the output predicted Y^p and the ground truth D^p, i.e. the correct label. The goal is to minimize the average error over a given training set consisting of prelabeled inputs Z^p. A widely used method for achieving this task is called gradient descent [LBBH98]. This method uses the gradient of the loss function to iteratively update the weights W:

$$W_k = W_{k-1} - \epsilon \frac{\partial E(W)}{\partial W} \tag{2.5}$$

To speed up the process the weights can be updated after every sample instead of after feeding the whole training set. This approach is called stochastic gradient algorithm or on-line update [LBBH98]. To efficiently compute the loss function the gradients are propagated from the output to the input [LBBH98]. A drawback of NN is that the learning procedure may get stuck in a local minimum [Vap99]. Further, the construction of the network is not part of the learning procedure since its sole task is to find a function which minimizes the error on the training set for a given network [Vap99]. Therefore, the size of the network has to be tuned by hand or heuristically to yield high accuracies and at the same time generalize well for unseen samples [MNY99].

Support Vector Machine

The SVM classifier is a binary classifier [MNY99]. Its aims is to find an hyperplane which optimally separates the training sample vectors [Vap99]. Optimal means that the distance between the closest vectors and the hyperplane is maximized [Vap99]. The hyperplane is constructed by a linear combination of the training sample vectors. The vectors with non-zero coefficients are referred to as *support vectors*. However, this approach only linearly separates the data [Vap99]. Therefore, to allow nonlinear decision boundaries a kernel function is used to map the training sample vectors to a high-dimensional space [Vap99]. Using a nonlinear kernel function a linear decision boundary in the kernel space then corresponds to a nonlinear decision boundary in the input space [Vap99]. Besides a linear kernel, frequently used nonlinear kernel functions are the polynomial kernel and the Radial Basis Function (RBF) kernel [MNY99]. The problem of non-separable vectors can be solved by using a soft margin which introduces a constant for controlling the tolerance to classification errors in the training stage [MNY99]. Due to the fact that the SVM classifier can only separate two classes, multiple SVMs have to be combined for the classification of more than two classes. An advantage of the SVM is that in contrast to NN the training of samples has a unique solution, i.e. a global minimum [Vap99]. A detailed explanation of SVMs is given in Section 3.2.3.

2.1.3 Digit Recognition Systems

In this section several state-of-the-art digit recognition systems are presented. However it is not intended to provide an extensive survey, but instead present exemplary approaches which deliver state-of-the-art performance. Extensive surveys to this topic can be found in [PS00], [AYV01] and [Sri06]. The performance of the algorithms is compared using results reported on the MNIST database which is explained in detail in Section 4.1.1.

This section is divided as follows: First, approaches are presented which use features encoding stroke directions. Second, two approaches which are based on the biological vision system are presented. Third, an approach which uses deformed image matching to compute the similarities between images is shown. Fourth, approaches which incorporate the feature extraction process into the classifier are described.

Stroke Directions

Liu et al. [LNSF04] present different approaches using local stroke directions of the handwritten digits as features. These features are extracted from contour chaincodes and gradient directions [LNSF04]. An advantage of those features is that they are robust with respect to stroke-width [CKLS07].

In this context, the contour of a character is defined as the outer boundary between the character and the background. A simple and frequently used algorithm for extracting it is the Square-tracing algorithm [CKLS07]. Based on a binary image a starting point, i.e. black pixel on the contour, is chosen. Following the simple rule of turning left at black pixels and turning right at white pixels the contour is build up by the visited black pixels. The algorithm is finished after visiting the starting point for a second time. The contour chaincode is created by assigning each contour pixel an 8-directional code [LNSF04]. To avoid the computation of the contour, Liu et al. [LNSF04] propose the use of a raster scan of the image to directly compute the chaincode [LNSF04]. The contour pixels are then subsequently assigned to corresponding direction planes (one plane for each of the 8 directions). For the extraction of gradient directions this is done similarly. However, instead of computing the contour, the x- and y-gradients (g_x, g_y) of the image are computed using horizontal and vertical Sobel filters, respectively. Then, the gradients are sampled to 8 directions using gradient decomposition, i.e. the vector is linearly decomposed into its two nearest directions. The results are again stored in direction planes. The final feature vector is computed by downsampling the feature planes using a Gaussian filter.

The classification is done using either a polynomial classifier or an SVM with an RBF kernel.

Using directional gradient features with 12 directions instead of 8, a Principle Component Analysis (PCA) to compress the feature vectors (to 160 dimensions) and a Modified Quadratic Discriminant Function (MQDF) classifier Liu et al. [LD05] were able to improve this approach and achieve an error rate of 0.46% evaluated on the MNIST database.

Vision-Based Features

Teow et al. [TL02] propose a feature extraction approach which is inspired by the biologic vision system. The proposed feature set is linearly separable even for a large set of training samples (60.000 samples from the MNIST database) [TL02]. They follow the traditional approach of separate modules for feature extraction and classification. In the first step, an intensity normalized image I^+ and its inversion $I^- = 1 - I^+$ are computed. This simulates the on- and off-channels in the visual system [TL02]. Secondly, 16 filters of size 5x5 designed to detect edges and end-stops of various orientations are used. These filters simulate the behaviour of receptive fields in the visual system [TL02]. The results of the convolution are fed to a truncated linear function which sets negative values to 0. The remaining non-negative numbers correspond to the strength of each feature [TL02]. To reduce the number of falsely detected corners and edges the result is multiplied with I^+ and I^-, respectively [TL02]. Moreover, to reduce the number of features, a magnitude-weighted average is computed for 9x9 overlapping windows. The feature vector is built up by 32 feature maps (16 for I^+ and I^-, respectively) of size 9x9, i.e. has a dimension of 2592. For classification the results of a single-layer perceptron NN, a linear multiclass SVM and a K-NN classifier are compared. They propose a triowise learning approach for the perceptron NN and the SVM. In contrast to the one-against-one training a one-against-two training with soft voting is proposed. Soft voting means, that the outputs of the classifiers are summed up without prior thresholding [TL02]. To additionally improve the results, slant correction is performed by computing a regression line through the centroid of the image. The best results on the MNIST database reported (error rate of 0.59%) are achieved using the linear SVM classifier [TL02]. This approach is similar to the Convolutional Neural Network (CNN) by Lecun et al. [LBBH98]. However, in a CNN the features are learned in the training phase and further the CNN has multiple layers which allows a non-linear decision boundary.

Another vision based approach is proposed by Lambusch et al. [LBM08]. They propose a feature extraction method based on learned sparse representations. The feature extraction is divided into two stages. First, coefficients are learned or computed and second, local extrema are used to obtain shift invariance. For the coefficients either PCA, Gabor wavelets or sparse coding is used. The local extrema are computed in 9x9 subregions. An SVM classifier with RBF kernel is used for classification. They achieve an error rate of 0.59% on the MNIST database. A detailed explanation of this approach is given in Section 3.2.

Deformation Model

Keysers et al. [KDGN07] propose an image matching approach for digit recognition. The main idea is to map the pixels of a test image onto the pixels of a reference image. The quality of the mapping is determined by a distance function which is used in combination with a k-NN classifier to predict the class label of the test image.

The proposed matching model is called Image Distortion Model (IDM). In contrast to other matching models, the IDM only includes absolute constraints [KDGN07], i.e. only absolute positions are considered - relations between neighboring pixels are neglected. Since minimization of the matching function can be done independently for each pixel this reduces its search space leading to a lower computational complexity of the algorithm compared to matching models with stricter constraints [KDGN07]. However this model is not able to retain intraclass variability [KDGN07]. To cope with this disadvantage Keysers et al. [KDGN07] propose the use of the local image context to improve the matching. More precisely, vertical and horizontal Sobel filters are used to compute the image derivatives. The local context is then subsequently formed by these values in a local 3x3 pixel environment. The local context is then represented by a feature vector of dimension 18, i.e. the vertical and horizontal filter responses for 9 local pixels.

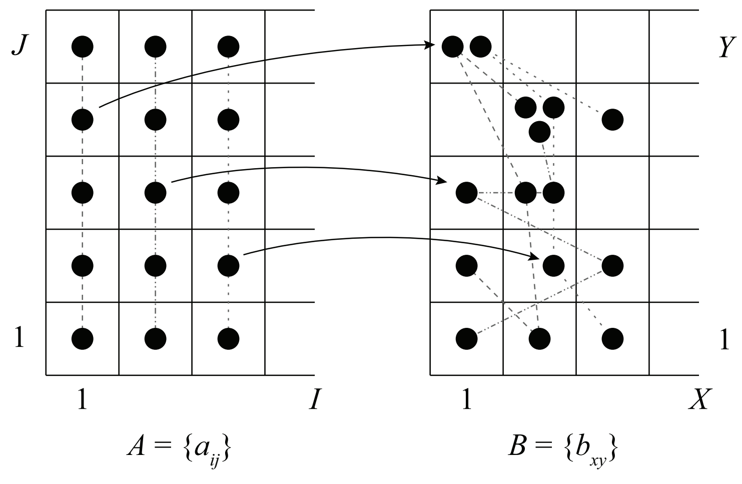


Figure 2.3: Matching example using the IDM. Each image pixel of the test image A is mapped onto a pixel in the reference image B [KDGN07].

Considering a test image $A = \{a_{ij}\}$ and a reference image $B = \{b_{xy}\}$ with pixel positions $(i, j), i = 1, ..., I, j = 1, ..., J$ and $(x, y), x = 1, ..., X, y = 1, ..., Y$ respectively, the distance function in the IDM is defined as:

$$d(A, B) = \sum_{i,j} \min_{(x_{ij}, y_{ij})} f(A, B, x_{ij}, y_{ij}) \tag{2.6}$$

with $x_{ij} \in \{1, \ldots, X\} \cap \{i' - w, \ldots, i' + w\}$ and $y_{ij} \in \{1, \ldots, Y\} \cap \{j' - w, \ldots, j' + w\}$. The parameter w denotes the warp size and the position (i', j') in the reference image is computed by linear mapping i and j to the range $[1, X]$ and $[1, Y]$, respectively. This means, that for every pixel a_{ij} the pixel with the minimal distance, with respect to some distance function f, is searched and the distances are accumulated. The computational speed-up gained in comparison to other matching models comes from the fact, that the minimum can be put inside the summation because the IDM allows for independent computations of the pixels. The search area is restricted by w which defines the maximum displacement allowed. The Euclidean distances of the local context vectors are used as the distance function f. The matching process is shown in Figure 2.3

For classification a K-NN classifier with $k = 3$ is used. For the MNIST database and the UPSP database error rates of 0.54% and 2.5% are achieved, respectively [KDGN07]. Although the computational effort is reduced to 0.5ms per distance computation [KDGN07], in comparison to 2-Dimennsional Warping with 11ms [KDGN07] for instance, it is still the major drawback of this approach. Due to the use of a K-NN classifier the classifaction of one test sample requires the computation of the distances to every sample in the training set. Considering the MNIST database which includes 60,000 training samples, for the classification of one test sample 60,000 distance computations are needed which sums up to a classification time of 30s. As one way to alleviate this problem Keyseers er al. [KDGN07] propose the use of learned prototypes (one for each class). However, this increases the error rate to 4.9%.

Convolutional Neural Networks

The drawbacks of the *classical* approach of a feature extraction module and a classification module are that first, the design of the feature extractor is cumbersome and time consuming and second, it is very specific to a certain task and cannot be reused without modifications [LBBH98]. As shown by Lecun et al. [LBBH98] it is possible to achieve state-of-the-art recognition accuracies without using a hand-crafted feature extractor but instead incorporate the feature extraction process in the (trainable) classifier.

Using the raw image data as features and a MLP classifier with one hidden layer and 300 neurons Lecun et al. [LBBH98] report an error rate of 4.5%. Adding a second hidden layer with 100 neurons the result improves to 3.5%. However, in both of these methods the layers are fully connected, i.e. all neurons of one layer are connected to all neurons of the following layer. This has two disadvantages. First, the network has no built-in shift, rotation or scale invariance [LBBH98]. This problem could be overcome by increasing the size of the network and using many training samples, however this would lead to parts in the system with duplicate weight structure [LBBH98]. Second, even though nearby pixels are highly correlated in images, this is not exploited since the connections are global and therefore the local structure of the image is not emphasized [LBBH98]. To overcome these drawbacks, Lecun et al. [LBBH98] propose the so-called CNN which uses weight sharing and local receptive fields.

The structure of a CNN is a *bi-pyramid*. In the first layer, 6 feature maps are used to extract local features of the image. Every pixel in the feature map is connected to a 5x5 area in the previous layer, i.e. the input image. The 25 weights used for this connection are shared over the whole feature map. This means, a feature map extracts the same local features at different locations. Due to the fact, that the 5x5 regions are overlapping, the operation of computing a feature map is similar to a convolution using the weight matrix as the kernel [LBBH98]. After the convolution stage, a subsampling stage follows. In

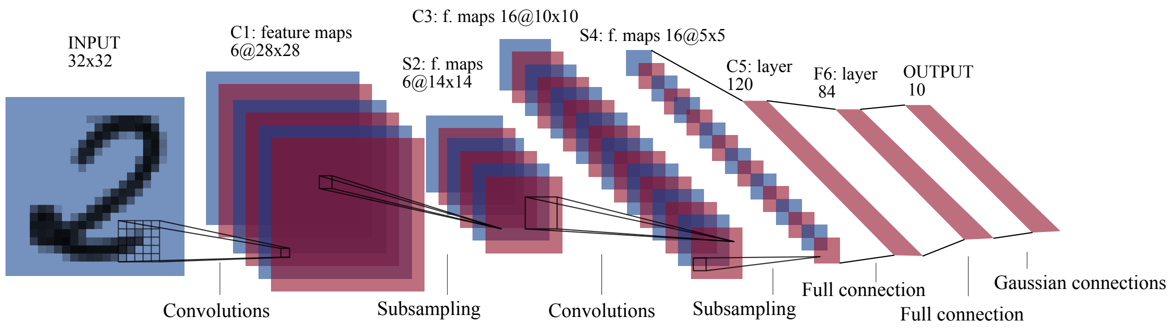


Figure 2.4: The different layers of the *bi-pyramid* in a Convolutional Neural Network [LBBH98].

this stage, each feature map is subsampled by averaging distinct 2x2 regions multiplying the result with a trainable coefficient and adding a trainable bias. The results are then passed to sigmoid functions, which have a linear and non-linear behavior depending on the weights. After the subsampling stage, another convolution stage follows and so on; hence, the name *bi-pyramid*. In every stage, the number of feature maps increases and the resolution decreases. Figure 2.4 depicts the different layers of the *bi-pyramid*.

Lecun et al. [LBBH98] achieve an error rate of 0.95% on the MNIST database. Ciresan et al. further improve this architecture by proposing so-called Deep NN [CMGS10]. This way and by the combination of multiple Deep NN [CMS12] they achieve error rates of 0.35% and 0.23%, respectively.

2.2 Document Analysis for Digit Recognition

In order to extract the fields containing the numerical data, i.e. the digits, these fields have to be located beforehand. The documents in scope of this report, i.e. the weather records, are forms which have been filled out by hand. Therefore, an underlying tabular structure is present in these documents. In this section, first approaches for detecting the basic elements of this structure, i.e. lines are presented. Second, recognition systems are shown which aim to recognize numerical data in documents.

2.2.1 Form Analysis

In forms the cells are defined by surrounding grid lines. Therefore, to extract the fields from forms, first the lines have to be localized. Further, even if the structure of forms suggests certain areas which are applicable for handwriting, handwritten strokes may cross lines. Therefore, the removal of lines without breaking handwritten strokes is important.

This section is divided as follows: First, methods for line detection and removal are presented. Second, since some of those methods rely on upright images and document images are often skewed - due to the scanning and printing process - approaches to cope with this problem are shown.

Line Detection and Removal

One way to detect lines in images is the use of vertical and horizontal projection profiles [LDW95]. Since vertical and horizontal projections are influenced by skew, the image must first be deskewed or the projection has to be performed at a skew angle [NB06, LDW95]. The lines are then located by searching for local maxima in the projection profile. To increase the signal-to-noise ratio Nielson et al. [NB06] propose to use a matched filter. This way the peaks are clearly separated and an appropriate threshold value is sufficient to detect the lines [NB06].

Using a vector representation of the image provides another way for line detection [ZLDP01]. Zheng et al. [ZLDP01] for instance propose the use of Directional Single-Connected Chain (DSCC) elements. A DSCC is built up by black pixel run-lengths which are single-connected. For instance, a horizontal DSCC is an array of connected vertical run-lengths. Frame lines are then constructed using multiple non-overlapping DSCC elements. To remove spurious lines and fill line breaks, form line constraints are introduced [ZLDP01].

Since the Hough line transform builds up a voting matrix of all possible lines in a binary image, it is frequently used to detect form lines [ZLDP01, Gar96]. The Hough matrix is constructed by iterating over every black pixel in an image and for each possible line through the pixel, which is represented by a pair (θ_i, ρ_j), a vote is cast [Gar96]. The parameters θ_i and ρ_j represent the coefficients in the normal representation of a line:

$$\rho_j = x \cos\theta_i + y \sin\theta_i \tag{2.7}$$

The relationship of a line in a Cartesian coordinate system and the corresponding coordinates in the Hough matrix are shown in Figure 2.5.

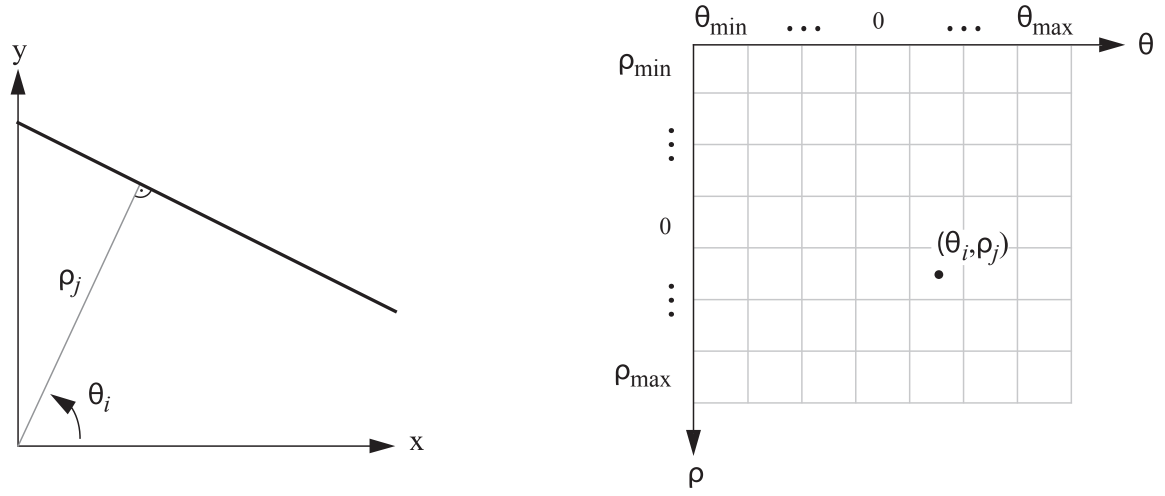


Figure 2.5: Line in a Cartesian coordinate system with its parameters in normal form (left) and the corresponding point in the Hough matrix (right) [Gar96].

Horizontal and vertical lines can now be found by searching for dominant lines, i.e.

maxima in the Hough matrix, in an appropriate angular range. However, the Hough matrix contains no information about the length of lines [Gar96]. Therefore, Garris et al. [Gar96] propose a method in which pixels on the trajectory of the dominant lines are counted to retrieve the frame lines.

For stroke-preserving line removal Garris el al. [Gar96] first estimate the width of a line. Considering each line as a sequence of slices, the median height (in case of horizontal lines) of these slices is computed. The slices with height less or equal to this value are then removed. However, this can lead to holes in characters touching a line [Gar96]. For instance, if the digit '0' touches a horizontal line it may break apart. This problem is treated by examining the contour of two character pieces. Using line projections of the contour the algorithm decides if the pieces compromise a curvature, i.e. the hole should be filled. Arvind et al. [AKR07] propose a method for the restoration of handwritten strokes which is based on the detection of diagonal strokes (crossing the line) and U strokes (touching the line). These patterns are detected using the thickness of the handwritten strokes and a set of rules. The holes are then filled using either Bresenham's algorithm for drawing straight lines or a step formation in case of U strokes [AKR07].

Skew Estimation

One method for estimating the skew angle is the use of projection profiles [CCMM98]. The projection profiles are computed along different angles and the skew angle is selected which optimizes an objective function. For instance Arvind et al. [AKR07] propose the use of the minimum entropy of the horizontal projection profiles. For an angular range of ± 10 degrees the image is rotated and the entropy E of the horizontal profiles H is calculated by:

$$E = \sum_i -H(i) * \log(H(i)) \tag{2.8}$$

The angle with minimal entropy is used as the skew angle and deskewing is performed by rotating the image in the opposite direction by that angle. To enhance the result, this is done separately for precomputed blocks in the image [AKR07].

Other methods use the Hough transform or nearest neighbor clustering [CCMM98]. A survey of different techniques for skew estimation is given in [CCMM98].

2.2.2 Digit Recognition in Documents

Earlier systems for recognizing handwritten digits in documents include for instance US addresses [SGS93] and census forms [GBC$^+$95], [Bre94], [GBC$^+$97].

More recently, Bulacu et al. [BBZS09] presented a system for recognizing handwritten digits in documents from the archive of the Cabinet of the Dutch Queen. The documents were recorded in the years 1798-1988 and large portions of them were written by a single writer. The documents from the year 1903 - all written by the same writer - were selected for evaluation. All of these documents can be referenced by page reference number, therefore recognizing numbers is crucial for indexing them [BBZS09]. Additionally, days and decision number (referencing the paragraphs) are recognized. For recognizing digit strings they employ a over-segmentation approach, i.e. a big number of possible cuts is evaluated using the class-member score of the classifier [BBZS09]. Further deslanting at a fixed 45°angle and the MNIST image normalization method is applied. The digit recognition is performed without feature extraction using either MLP or RBF NN. To enhance the accuracy, random elastic morphing is used to generate additional synthetic training samples to cope with intraclass variations. This way, the robustness of the NN is increased reducing the error rate on the MNIST database from 1.4% to 0.9% [BBZS09]. Additional improvement is achieved using a lexicon which improves the recognition accuracy up to 7.9% on the actual documents [BBZS09]. The overall accuracy for the recognition of days, decision numbers and page references (whole digit strings) is 90.9% and 97.3% without and with the use of a lexicon, respectively [BBZS09]. The evaluation was conducted using 6740 manually extracted fields.

Richarz et al. [RVF12] propose a semi-supervised method for the transcription of historic weather documents. These documents were recorded by the German Weather Service between the years 1877 and 1999. For preprocessing, they use Otsu's method for binarization [RVF12]. Line recognition is done using the Hough transform because in contrast to projection profiles this does not require rotation corrected images [RVF12]. The peaks in the Hough parameter space are found using sliding windows. The average and the standard derivation in the sliding windows are used to compute a local threshold. Non-maximum suppression is applied to reduce noise. To remove spurious lines, the fact that the forms contain only horizontal and vertical lines, i.e. orthogonal lines, is used to filter the results using an angular histogram. To determine the tabular structure of the form the intersection points of the lines are clustered. The extracted characters - the alphabet includes the digits 0-9, the german abbreviations for the cardinal directions N, S, W, O and the characters T, I and L, are normalized similarly to the MNIST database. For feature extraction they use 4 different representations, namely PCA features, dimensionality reduced features using an autoencoder network and features extracted using a non-negative matrix factorization [RVF12]. The semi-supervised learning approach is divided into four steps:

1. *Clustering:* The samples are clustered into 50 clusters for each representation using k-means clustering.

2. *Labeling:* The clusters are labeled by hand. This means, for each feature representation 50 clusters have to be labeled which sums up to a total number of 200 labeling operations.

3. *Voting:* To reduce errors in the labeling process, the results from the different feature representations are combined using a voting scheme.

4. *Training:* The labeled training samples are used to train either a K-NN, Relevance Vector Machine (RVM) or MLP classifier.

The evaluation is conducted on a dataset of 58 documents using a 6-fold cross validation scheme. The best results reported (86.9% accuracy) were achieved using a K-NN classifier with $k = 3$ [RVF12]. To further improve the results, they incorporate prior knowledge about the extracted fields. This information is stored in a template and helps to restrict the possible classes, i.e. only numbers or only cardinal directions etc. This way, the accuracy is increased to 89.0% [RVF12]. To measure the impact of the semi-supervised learning approach, the same experiments are additionally conducted using prelabeled data. Interestingly, even though in this case 6,550 manual labeling operations are needed the accuracy is not increased [RVF12]. This hints, that the proposed approach can achieve the same accuracy with only a fraction of manual labeling operations required.

Palacois et al. [PG08] propose a system for processing handwritten bank checks; specifically, the courtesy amount. In the first step, they use a binary image to compute Connected Components (CCs). Using a set of rules they are able to locate the courtesy amount. The handwritten number is then segmented using a so-called *Drop Fall* approach [PG08]. After a normalization step, which includes slant, size and stroke thickness normalization, the digits are classified using multiple (3 or 4) MLP neural networks. They compare different strategies for combining the results of those networks. Due to the problems arising from falsely recognized numbers on bank checks the system can be adjusted to achieve a low false-positive rate [PG08]. This, however, increases the rejection rate [PG08]. For the NIST database a correct classification rate of 92.7% with a rejection rate of 5.7% is achieved [PG08]. For evaluations conducted with U.S. checks they report a correct recognition rate of 47% with a rejection rate of 45% [PG08]. However, this includes all stages of the system from an image of a check to the recognition of the whole courtesy amount string.

Summary

In this chapter first popular classifiers for digit recognition were presented. Although the K-NN classifier delivers fairly high classification performance, the parameter k has to be chosen carefully and its biggest drawback is the high computational costs. Moreover, although SVMs and NN are conceptional similar, they differ in the training process. By searching for a decision boundary with maximal margin instead of any boundary, SVMs generalize better than NN. Furthermore, for NN the size of the network is crucial and it has to be either tuned by hand or using heuristics. However, as the CNN shows, NN can be suited to specialized tasks as for instance incorporating the feature extraction process into the classifier.

The digit recognition systems presented were divided into four different categories: firstly, the methods based on stroke directions bear the advantage, that they are robust with respect to the stroke-width. The best performance in this category is achieved using gradient directions. Secondly, vision-based methods use filterbanks simulating the behavior of receptive fields in the vision system for feature extraction, which leads to similar performance. Although CNNs are based on the same features, they incorporate the feature extraction process into the classifier which has the advantage that the filters are trained simultaneously with the rest of the network. However, this leads to a complex system with multiple hidden layers. In contrast to these methods, the IDM approach is based on mapping the pixels of a test image onto the pixels of a reference image. As this approach includes the computation of a distance function, this additionally allows the identification of similar images. However, due to the use of a K-NN classifier the biggest drawback of this approach is its computational costs. All approaches presented deliver accuracies of 99% or more on the MNIST database.

Moreover, methods for form analysis were described. It was shown, that lines can either be detected using vector representations combined with heuristics, projection profiles or the Hough transform. Even though projection profiles provide simple and effective means for this task, skew detection has to be performed beforehand. However, it is the only approach directly applicable for grayscale images as both the vector representation and the Hough transform are based on binary images.

Finally, entire systems for the recognition of handwritten digits in historic documents, weather reports and bank checks were depicted. It was shown, that although the digit recognition accuracy on digit databases like MNIST reach accuracies of 99% and more, real world scenarios are still challenging. For instance, even though Bulacu et al. [BBZS09] achieve an accuracy of 99.1% on the MNIST database, the dataset used for evaluation is from a single writer and the segmentation was done manually, the recognition accuracy on the documents from the archive of the Cabinet of the Dutch Queen is 90.9% without the use of a lexicon. However, it has to be noted, that these results are achieved on digit strings. Similarly, Richarz et al. [RVF12] achieve an accuracy of 89.0% per digit for the historic weather documents in scope of their work. Moreover, the depicted system for processing handwritten bank checks achieves a correct recognition rate of 47% even though 45% of the checks are rejected.

Chapter 3

Methodology

In this chapter the steps for extracting the handwritten temperature information from scanned weather records are described. The starting point are RGB images of weather records which have been scanned at a resolution of 300dpi, i.e. the same resolution used for the MNIST, USPS and CVL handwritten digit databases (see Section 4.1.1). In the first step, the layout analysis, the images are preprocessed and the form structure is reconstructed. This enables the extraction of the Region of Interest (ROI), i.e. the region containing the temperature values. In this region, after removing the horizontal and vertical form lines, the symbol cells are extracted. In the second step, the feature extraction, starting from normalized digit images, features are extracted using trained PCA basis functions. Finally, in the classification, an SVM is used to predict the labels of the digits and signs.

This chapter is divided as follows: first, in Section 3.1 the methods used for extracting the digit and signs from the weather records is explained. Second, the digit recognition approach used is described in Section 3.2.

3.1 Document Analysis

Starting from scanned images of weather records the goal of the document analysis is to locate the areas containing the temperature measurements, i.e. the numerical data, and then extract the digits and signs for further processing. The process is divided into several steps:

Rotation Correction
Due to the scanning and printing process the documents may not be aligned upright. However, this is necessary for the horizontal and vertical line detection and further it simplifies the description of the form by aligning the lines with the coordinate axes. This is achieved using an orientation histogram of the gradient vectors.

Table Structure Reconstruction
The main structure of the weather records regarded in this report is a table containing measurement values. Reconstructing this tabular structure enables the localization of the ROI and the segmentation of the digits and signs. The table reconstruction is based on the detection of the lines separating the table cells.

ROI Localization

In order to extract the temperature measurements, the region containing this data must be located beforehand, i.e. the ROI. This is done using a-priori information about the form in combination with the reconstructed table.

Line Removal

To segment the single digits and signs the form lines are removed. However, part of the symbols may cross the lines. To preserve the characteristics of the digits, a line removal preserving handwritten strokes is applied.

Segmentation

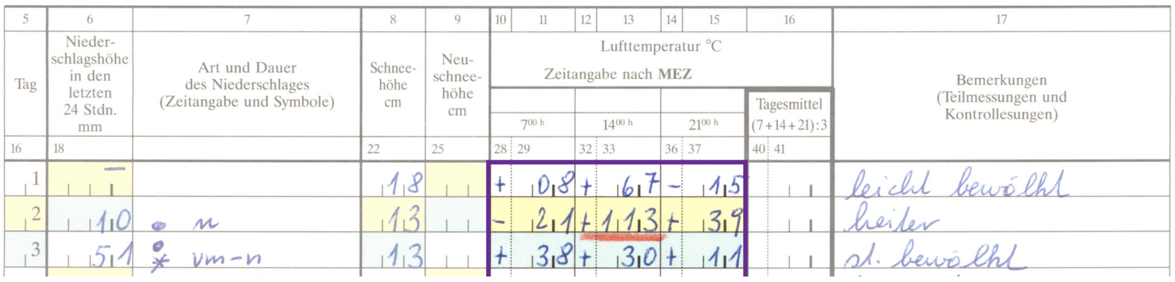
In the last step the pixels belonging to digits and signs have to be grouped and extracted. This is done using local thresholds and CC analysis.

This section is divided as follows: first, in Section 3.1.1 the structure of the form and the a-priori information used is described. Then in Section 3.1.2 the method used for correcting the rotation of documents is shown. The reconstruction of the tabular structure is presented in Section 3.1.3 and the localization of the ROI is subsequently explained in Section 3.1.4. In Section 3.1.5 the stroke-preserving line removal approach is shown. Finally, in Section 3.1.6 the remaining steps for extracting the digits and signs are described.

3.1.1 Form Structure

The weather records regarded in this report are divided into an upper part with general information about the measurement station and the month of the recording and a lower part containing the measurement values stored in a table. The table has eight main columns: day of the month, height of precipitation, type and duration of the precipitation, depth of snow, depth of fresh snow, temperature values, remarks. The column containing the temperature values is further divided into four columns with the measurements of morning, noon, evening and mean value. Rows in the table correspond to the days of a month. The ROI is defined as the area containing the three temperature measurements (three columns) for all days of a month (fixed to 31 rows). Beside this global structure the temperature cells are locally divided into sign cells and cells for the digits left of and right of the decimal point. The header of the table and the ROI are shown in Figure 3.1.

A-priori information about the form is stored in so-called *Form Properties*. On the one hand this allows corrections to the reconstructed tabular structure and on the other hand this information is used to detect the ROI and the local cells containing digits and signs. Further, using variable properties instead of fixed values allows an adaption of the system to other weather records with similar structure. The following *Form Properties* (see Figure 3.2) are used: the approximate vertical distance from the top, approximate width and height of columns and rows in the ROI, relative widths of the three temperature columns, line patterns of the frame lines, relative positions of the local dividers separating the cells, types of the local cells (sign or digit).



5	6	7	8	9	10 11	12 13	14 15	16	17
Tag	Nieder-schlagshöhe in den letzten 24 Stdn. mm	Art und Dauer des Niederschlages (Zeitangabe und Symbole)	Schnee-höhe cm	Neu-schnee-höhe cm	Lufttemperatur °C Zeitangabe nach **MEZ** 7⁰⁰ h	14⁰⁰ h	21⁰⁰ h	Tagesmittel (7+14+21):3	Bemerkungen (Teilmessungen und Kontrollesungen)
16	18		22	25	28 29	32 33	36 37	40 41	
1	–		1,8		+ 0,8	+ 6,7	− 1,5		leicht bewölkt
2	1,0	◉ n	1,3		− 2,1	+ 11,3	+ 3,9		heiter
3	5,1	● ✱ vm-n	1,3		+ 3,8	+ 3,0	+ 1,1		st. bewölkt

Figure 3.1: Header of the table containing the measurements in the weather records. The area holding the temperature values, i.e. the ROI is highlighted.

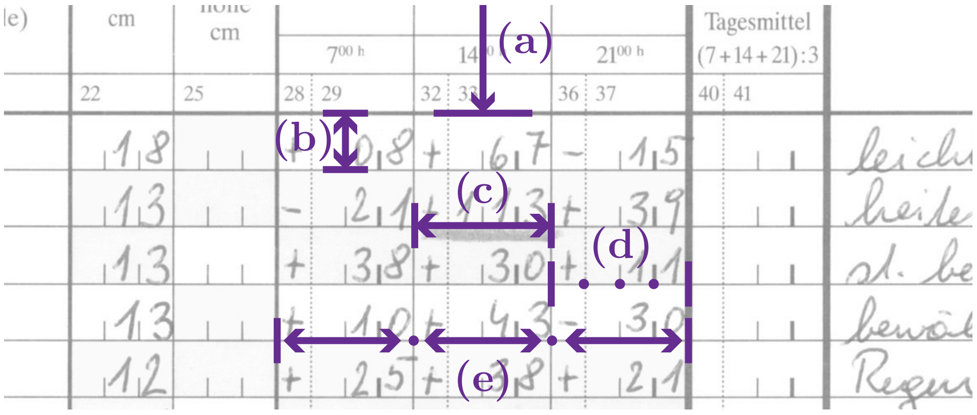


Figure 3.2: Visualization of the *Form Properties* ROI Position (a), Average Cell Height (b), Average Cell Width (c), Local Line Pattern (d) and Relative Cell Widths (e).

3.1.2 Rotation correction

The rotation correction is based on the skew estimation approach proposed by Diem et al. [DKS12]. They propose the use of an orientation histogram of the gradient magnitudes to estimate the rotation angle. The gradients f_x and f_y are defined as

$$f_x(x, y) = f(x + 1, y) - f(x - 1, y) \tag{3.1}$$

$$f_y(x, y) = f(x, y + 1) - f(x, y - 1) \tag{3.2}$$

where the input image is denoted with $f(x, y)$. This is equivalent to filtering the input image with the kernel h_x = [-1 0 1] and its transpose $h_y = h_x^T$ for f_x and f_y, respectively. To make the gradient robust to noise Gaussian filtering with $\sigma = 3$ is performed beforehand. The filter size n is chosen to support ±3 standard derivations of the mean:

$$n = \lceil 6\sigma \rceil + 1 \tag{3.3}$$

The magnitude $m(x, y)$ and orientation $\theta(x, y)$ of the gradient vectors are computed using

$$m(x, y) = \sqrt{f_x(x, y)^2 + f_y(x, y)^2} \tag{3.4}$$

$$\theta(x, y) = \tan^{-1} \frac{f_y(x, y)}{f_x(x, y)}, \qquad \theta(x, y) \in [-\pi/2, \pi/2] \tag{3.5}$$

In the next step a gradient orientation histogram is built. For each bin in the histogram, i.e. interval of gradient orientations, the gradient magnitudes $m(x, y)$ for all pixels (x, y) with gradient orientation $\theta(x, y)$ falling into the interval are accumulated. That means, the pixels contribution is weighted by its gradient magnitude. Therefore, pixels belonging to edges (strokes and table lines) are emphasized over pixels on uniform surfaces (as for instance background). Further, the accumulated values are normalized, i.e. mapped into a $[0, 1]$ interval. The orientation histogram for an example is shown in Figure 3.3. The orientation of the image is assumed to be in the interval $[-45°, 45°]$ and using a bin-size of $1°$ leads to 90 bins in the histogram.

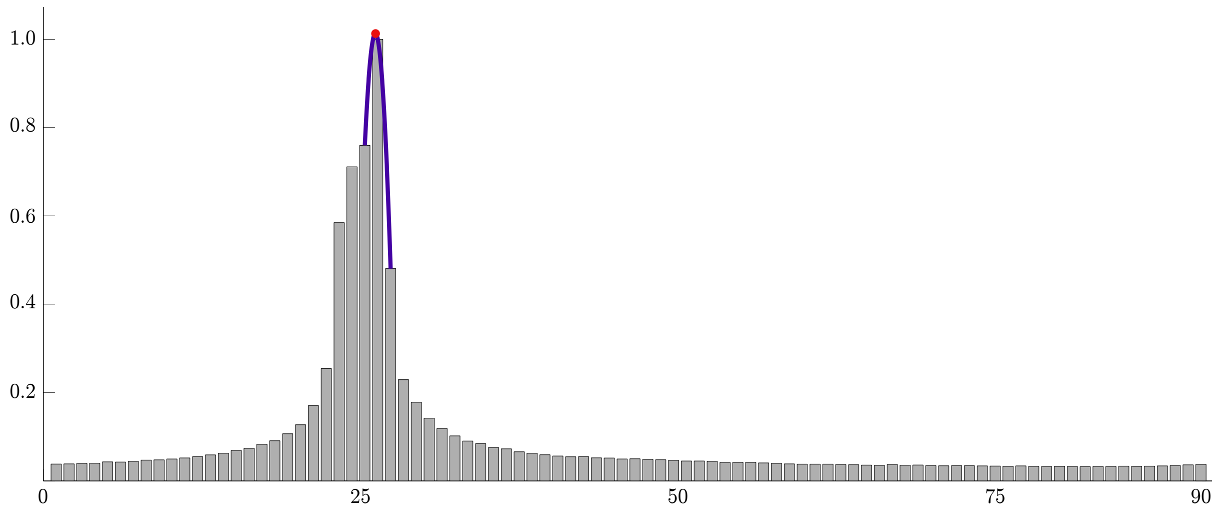


Figure 3.3: Orientation histogram of an example image. The x-axis and y-axis denote the orientation angle in degree and the normalized gradient magnitude, respectively. The second order polynom used for orientation estimation is shown in purple with its peak in red. The main orientation in this example is about 25°.

The orientation is first roughly estimated by searching for maximal bins h_i in the histogram. To suppress non-significant maxima a Gaussian filter with $\sigma = 3$ is used. Having detected the global maximum, in a $\pm\sigma/2$ neighborhood the corresponding local peaks are then found in the unfiltered histogram. In order to favor narrow peaks over broad peaks, the peaks are sorted by the difference of their value and the median of the neighboring bins:

$$\Delta h_i = h_i - \text{median}(h_{j \in \{i-5,\ldots,i,\ldots,i+5\}}) \qquad (3.6)$$

The maximum peak computed this way $h_{\max}$ is then further refined by fitting a second order polynomial:

$$p(x) = a_2 x^2 + a_1 x + a_0 \qquad (3.7)$$

The coefficients a_j are computed using the bins $h_{\max}, h_{\max-1}, h_{\max+1}$ as supporting points by solving a system of three linear equations. However, when using a high number of bins (¿ 10,000) the solution can become numerically instable due to the contrast of high values for the x-coordinates and low values (in interval $[0, 1]$) for the y-coordinates. This is avoided by transforming the x-coordinates of the supporting points to the interval

$[-1, 1]$. The maximum of the polynomial is found using the roots of the first derivative of $p(x)$.

3.1.3 Table Structure Reconstruction

The first step for reconstructing the table structure is to find vertical and horizontal lines in the image. Starting from the rotation corrected grayscale image vertical and horizontal projections are used to find the vertical and horizontal lines, respectively. These projections are computed using the accumulated column and row values of the grayscale image. As shown in Figure 3.4 the peaks in the vertical projection correspond to the vertical lines in the image.

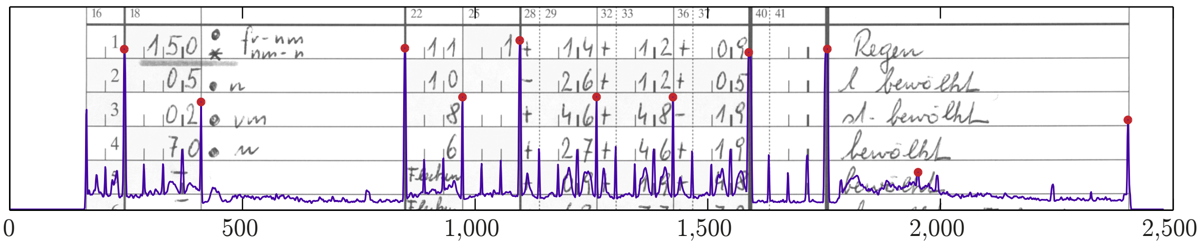


Figure 3.4: Image showing the vertical projection profile with the detected peaks and a part of the image in background (pixel rows 726 to 1133). The x-coordinates represent the image row and the y-coordinates are scaled to provide a meaningful visualization. The weather record used is from January 1994 from the measurement station Nasswald.

In order to detect these peaks, first they are identified globally and sorted in descending order. Peaks below a certain threshold are omitted. Secondly, to ignore smaller peaks in the neighborhood of larger peaks a minimum distance is introduced as a constraint. The constraint is enforced by starting with the peak with the largest value and subsequently ignoring all peaks in the neighborhood defined by the minimum distance. The red circles in Figure 3.4 depict the peaks detected which are used for further processing. Figure 3.5 shows the horizontal projection profile. The 32 evenly distributed peaks in the center correspond to the table lines separating the 31 rows representing the days of a month.

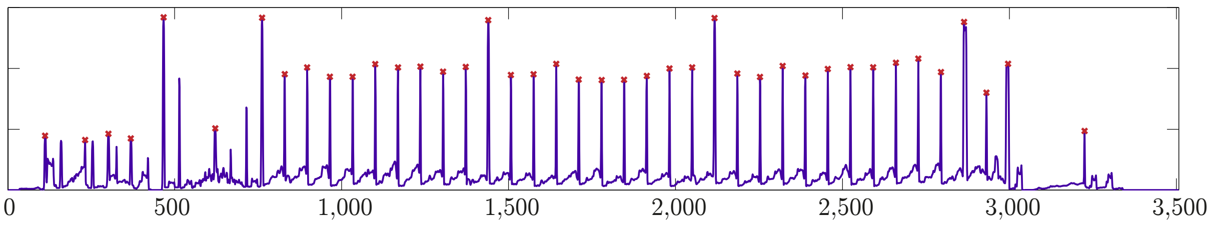


Figure 3.5: Image showing the horizontal projection profile and the detected peaks. The x-coordinates represents the image row and the y-coordinates the accumulated row values. The weather record used is from January 1994 from the measurement station Nasswald.

However, there are cases in which this method fails. Therefore, as a second approach the maximal vertical and horizontal black pixel runs are used as fallback. In this case the image is binarized beforehand using the method described in Section 3.1.6.

The reconstruction of missing horizontal lines is based on the assumption that if a line is missing the vertical difference between two consecutive lines Δy is significantly (more than 20%) larger than the *Average Cell Height* h. The number of missing lines n is then computed using:

$$n = \lfloor \frac{\Delta y}{h} + \frac{1}{2} \rfloor \tag{3.8}$$

Finally, the missing horizontal lines are evenly distributed in the area between the two detected consecutive lines.

3.1.4 ROI Localization

As mentioned previously, the ROI is defined as the area containing the three temperature measurements for all days of a month. The localization of the ROI is done using the reconstructed table, represented by horizontal and vertical lines, and the *Form Properties*. In a first step the frame lines near the image borders are removed. This is necessary due to peaks in the projection profiles in these areas resulting from black borders produced by the scanning process. These margins are defined as 3% of the number of columns and rows, respectively.

The vertical position and extend of the ROI is defined by the first horizontal frame lines below the ROI Position and the next 31 horizontal frame lines. The localization of the vertical position and extend is done by searching for consecutive cell-widths satisfying the *Relative Cell Widths*. To allow a certain degree of variation margins of $\pm 5\%$ are used. To ensure that no parts of the ROI are cut off, the area is extended by 10 pixels on all sides.

3.1.5 Line removal

To segment the single digits and signs it is necessary to remove the form lines. However, due to cursively written digits and signs, part of the symbols may cross the borders. To preserve their characteristics, a line removal preserving handwritten strokes is applied.

One possibility to achieve this is to compute the gradient images f_x and f_y, remove the horizontal and vertical lines, respectively and then integrate the gradient images to reconstruct the image. However, this leads to visible artifacts especially in areas where handwritten strokes cross the form borders. For example, in Figure 3.6 (f) in the area on the left, where the vertical borders were removed the background is no longer clearly separated from the handwritten stroke. Additionally, visible horizontal line patterns are introduced by this approach (see Figure 3.6 (f)).

The solution proposed is to use a Wiener filter to deconvolute the gradient images. The Wiener filter is a reconstruction filter. This means, starting from a filtered - or degraded - image $G(u, v)$, which was filtered with a known filter function $H(u, v)$, the task is to reproduce the original unfiltered image $F(u, v)$. The simplest approach to reproduce the

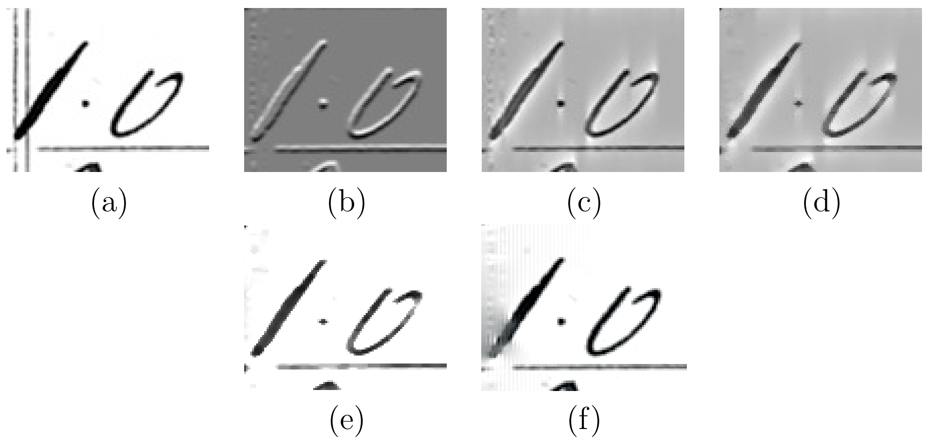


Figure 3.6: The five states of border removal using the Wiener filter (a-e). The original image (a), gradient image f_x (b), gradient after deconvolution (c), median filtered deconvolution (d) and resulting image after scaling the intensity values (e). For comparison, (f) shows the result of removing the vertical borders in the gradient image f_x and using gradient image integration for reconstruction.

original image is inverse filtering [GW08]. The reproduced image $\hat{F}(u,v)$ is constructed by dividing the filtered image $G(u,v)$ by the filter function $H(u,v)$ in the frequency space:

$$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)} \tag{3.9}$$

To cope with noise, the Wiener filter extends the inverse filtering approach by modelling the noise and the image as two random variables. The noise is considered uncorrelated to the image [GW08]. The Wiener filter minimizes the mean square error between the reconstructed image $\hat{F}(u,v)$ and the original image $F(u,v)$ [GW08]. This minimum is given in the frequency domain as

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K}\right] G(u,v) \tag{3.10}$$

where $|H(u,v)|^2$ is defined as the product of the filter function with its complex conjugate. The parameter K represents the signal-to-noise ratio, i.e. the power spectrum of the noise $|N(u,v)|^2$ divided by the power spectrum of the original image $|F(u,v)|^2$. The term in the brackets is called the Wiener filter.

Returning to the original problem of removing the image borders without removing the handwritten strokes, the parameter K can be used to limit the reconstruction effect of the Wiener filter. By setting $K = 0$ in Equation 3.10 the term in the bracket becomes $1/H(u,v)$ which is simply the inverse filter as defined in Equation 3.9. Due to the fact that K is constant over all frequencies, i.e. frequency independent, increasing it, limits the influence of the frequency dependent part of the inverse filter. Additionally, by computing the gradients the constant terms of the images are lost which are in case of the gradients f_x and f_y the vectors of accumulated row and column values, respectively. This means, that for instance after filtering with a vertical $[-1, 1]$ edge filter and ideal reconstruction (with

$K = 0$), the columns of the image which contained many foreground pixels (dark pixels) are brightened. In case of border removal, this is advantageous because the vertical and horizontal lines can be removed by these means. By using a Sobel filter, the gradients are not only dependent on one image row or column and therefore in the final result after reconstruction the stripes due to the missing constant are not noticeable any more. The horizontal and vertical Sobel filter kernels are shown in Figure 3.7 (left) and (right), respectively.

1	0	-1
2	0	-2
1	0	-1

1	2	1
0	0	0
-1	-2	-1

Figure 3.7: Sobel filter kernels for computing the horizontal (left) and vertical (right) gradients f_x and f_y, respectively.

The best results for the task of border removal were achieved using a Wiener filter with $K = 2$. In Figure 3.6 the whole process is visualized on an example. In order to remove the vertical lines in the original image (a), first the horizontal gradient is computed. As shown in (b) this removes the vertical lines. After filtering with a Gaussian filter with $\sigma = 0.5$, the Wiener filter is used to reconstruct the digits and signs. As shown in Figure 3.6 (c) the inner area of the digit '1' is reconstructed while the vertical lines remain removed. After that, a median filter with a size of 3x5 (in case of horizontal line removal the size is 5x3) is used to remove artifacts (see (d)). Finally, intensity scaling is applied to enhance the results:

$$I_s(x, y) = \begin{cases} I(x, y)^3 & \text{if } I(x, y)^3 < 1 \\ 1 & \text{else} \end{cases} \tag{3.11}$$

with the intensity $I(x, y)$ at the position (x, y) after reconstruction and filtering and $I_s(x, y)$ depicting the scaled intensities. In Figure 3.6 (e) the final result after intensity scaling is shown on an example. However, only the regions affected by vertical and horizontal lines have to be altered. This means, in the final image, only these areas are replaced while the rest of the image stays the same. This way, regions not affected by horizontal or vertical lines retain the initial image quality. The precise local segmentation of lines is done using row and column images with the vertical and horizontal gradients, respectively, in a search area defined by the reconstructed form. Using again projection profiles - this time of the absolute gradient image values - the largest n peaks are localized. The number of peaks n is defined by the line patterns. In case of single-lined borders two peaks are present (left and right of the line), in case of double-lined borders four peaks are present etc. In Figure 3.8 the search areas are shown for one row of the ROI.

However, the stroke-preserving line removal is not suitable for dashed lines. Therefore, the local cell lines separating the signs and digit are simply removed by replacing the localized lines with background pixels (white).

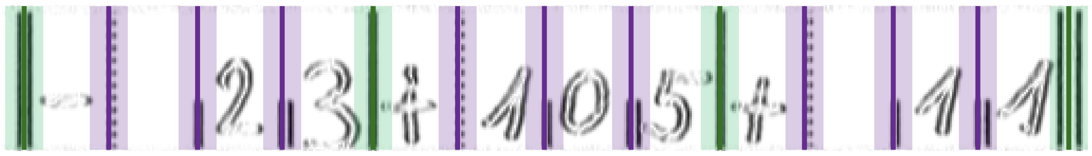


Figure 3.8: Absolute horizontal gradient image $|f_x|$ for one row in the ROI with the search areas for the precise frame line localization defined by the global (green) and local (purple) lines in the reconstructed form and a ±9 pixel margin.

3.1.6 Segmentation

In the next step the background is removed using a threshold image. The threshold image is computed by applying an adapted version of the Salvakis filter [Sav98]. In distinct 15x15 blocks the image pixels are grouped into foreground pixels v_i and background pixels w_j using a precomputed threshold obtained by Otsu's [Ots79] method, i.e. by separating the foreground and background pixels so that their intraclass variability is minimal. The local threshold t is then defined as the boundary separating the mean of the intensity values of the foreground pixels $\overline{v}$ and the mean of the intensity values of the background pixels $\overline{w}$:

$$t = \frac{\overline{v} + \overline{w}}{2}, \quad \text{with } \overline{v} = \frac{1}{I}\sum_{i=1}^{I} v_i \text{ and } \overline{w} = \frac{1}{J}\sum_{j=1}^{J} v_j \tag{3.12}$$

In Figure 3.9 this approach is compared to two state-of-the-art binarization approaches. Due to the high image quality of the weather records, this approach shows less missing character strokes and clearer frame boarders. The threshold image is then used to remove the background pixels. Additionally, a 5x5 Gaussian filter with $\sigma = 0.5$ is applied to enhance the result.

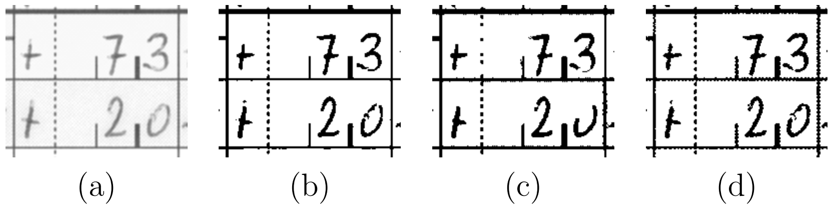


Figure 3.9: Comparison of different binarization techniques on an example (a). The method proposed (b) and approaches by Su et al. from the years 2010 [SLT10] (c) and 2011 [SLT11] (d).

The digits and signs are then extracted by computing CCs, i.e. 8-connected pixel regions. The centroids of the CCs are used to assigning them to the nearest cell centers.

The last step before the character recognition is a size normalization. The normalization method used is a moment-based aspect ratio preserving normalization [LNSF04] and is similar to the approach proposed by the MNIST database - in Section 4.1.1 this database is described in detail. The image is scaled to 20x20 by preserving the aspect ratio and then the centroid of the image is placed on the center of a standard plane of size 28x28. The scaled image is computed using bicubic interpolation. Depending on the values stored in the *Local Cell Types* the cells are labeled as signs or digits for further processing.

3.2 Digit Recognition

Starting from normalized digit images (or signs) the goal of the digit recognition module is to predict the corresponding class label. This is achieved using a supervised learning approach. First, in a training phase a set of images with annotated class labels are learned by the algorithm. This way the classifier adapts to the problem at hand and develops a decision boundary separating the different classes in the feature space in order to subsequently predict the class labels of new unseen sample images.

The digit recognition approach presented here is based on the work of Labusch et al. [LBM08]. In a first step, a PCA basis is trained using randomly placed image patches. This PCA basis is then used as a filter bank to extract the features from the digit images. To allow local shifts, extrema in non-overlapping regions are computed. Finally, an SVM is used as a classifier.

This section is divided as follows: first, in Section 3.2.1 the calculation of the PCA basis is explained. Second, the feature extraction process is depicted in Section 3.2.2 and finally the SVM classifier is described in Section 3.2.3.

3.2.1 Calculating the Basis Functions

The basis functions are learned using random image patches. As proposed in [LBM08] the size of the image patches is set to 13x13. A patch $P(x, y)$ is extracted by placing it at a random position in a training image. The training images used for this extraction are evenly distributed over the whole training set. As stated by Labusch et al. [LBM08] preprocessing is necessary for the PCA. This is due to the fact, that the constant term of the images, i.e. the mean, has to be removed in order for the PCA to provide meaningful results. This is achieved by converting the extracted patches into so-called *centered vectors* [LBM08]. First, for each patch $P(x, y)$ the mean pixel value $\overline{P}$ of the patch is subtracted:

$$S(x, y) = P(x, y) - \overline{P} \tag{3.13}$$

Next, $\overline{S(x, y)}$ the mean value over $\underline{\text{all}}$ $S(x, y)$ is computed. Finally, the centered vectors $S_p(x, y)$ are obtained by subtracting $\overline{S(x, y)}$ from all $S(x, y)$:

$$S_p(x, y) = S(x, y) - \overline{S(x, y)} \tag{3.14}$$

After extracting the patches, a PCA is used to learn the basis functions from these samples, i.e. the parameter of the underlying model. This is done by creating the covariance matrix and computing the eigenvectors and the corresponding eigenvalues.

Principle Component Analysis

The goal of the PCA is to find an orthonormal basis that transforms the data to a new coordinate system in which the variance of the projected data in each D-dimensional subspace is maximal [Bis06]. That means, in the first coordinate lies the greatest variance of the data, in the second coordinate lies the second greatest variance, etc. Assuming a set of K-dimensional samples $\boldsymbol{x}_n$ with $n = 1 \ldots N$, each sample can be represented by a linear combination of basis vectors $\boldsymbol{u}_i$:

$$\boldsymbol{x}_n = \sum_{i=1}^{K} \alpha_{ni} \boldsymbol{u}_i \tag{3.15}$$

where $i = 1 \ldots K$. Considering a projection to a 1-dimensional space using one basis vector $\boldsymbol{u}_1$ the variance is given by [Bis06]:

$$\frac{1}{N-1} \sum_{n=1}^{N} (\boldsymbol{u}_1^T \boldsymbol{x}_n - \boldsymbol{u}_1^T \overline{\boldsymbol{x}})^2 = \boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1 \tag{3.16}$$

with $\boldsymbol{S}$ being the covariance matrix with size $K \times K$:

$$\boldsymbol{S} = \frac{1}{N-1} \sum_{n=1}^{N} (\boldsymbol{x}_n - \overline{\boldsymbol{x}})(\boldsymbol{x}_n - \overline{\boldsymbol{x}})^T \tag{3.17}$$

where the expectation value of the samples $\overline{\boldsymbol{x}}$, i.e. the mean, is estimated by:

$$\overline{\boldsymbol{x}} = \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{x}_n \tag{3.18}$$

By introducing a Lagrange multiplier λ_1 to enforce a length of 1 for the basis vectors the basis vector maximizing the variance is found by setting the first derivative in respect to $\boldsymbol{u}_1$ equal to zero [Bis06]:

$$\boldsymbol{S}\boldsymbol{u}_1 - \lambda_1 \boldsymbol{u}_1 = 0 \tag{3.19}$$

That means $\boldsymbol{u}_1$ is an eigenvector of the covariance matrix $\boldsymbol{S}$. The above formula can be rewritten as:

$$\boldsymbol{u}_1^T \boldsymbol{S} \boldsymbol{u}_1 = \lambda_1 \tag{3.20}$$

which shows that the maximal variance is in direction of the eigenvector $\boldsymbol{u}_1$ having the largest eigenvalue λ_1, i.e. the first principal component. Additional basis vectors $\boldsymbol{u}_i$ are selected by sorting the eigenvectors by their corresponding descending eigenvalues λ_i. The first and second principle components for a set of 2-dimensional sample points are shown in Figure 3.10.

Returning to the original problem, the 13x13 image patches are reshaped to vectors with a dimensionality of $K = 169$. Similarly, after computing the PCA, the resulting basis vectors $\boldsymbol{u}_i$ of size K are reshaped to 13x13 and stored in a filter bank with a total number K filters. In Figure 3.11 examples of random image patches and PCA basis functions in the filterbank are depicted.

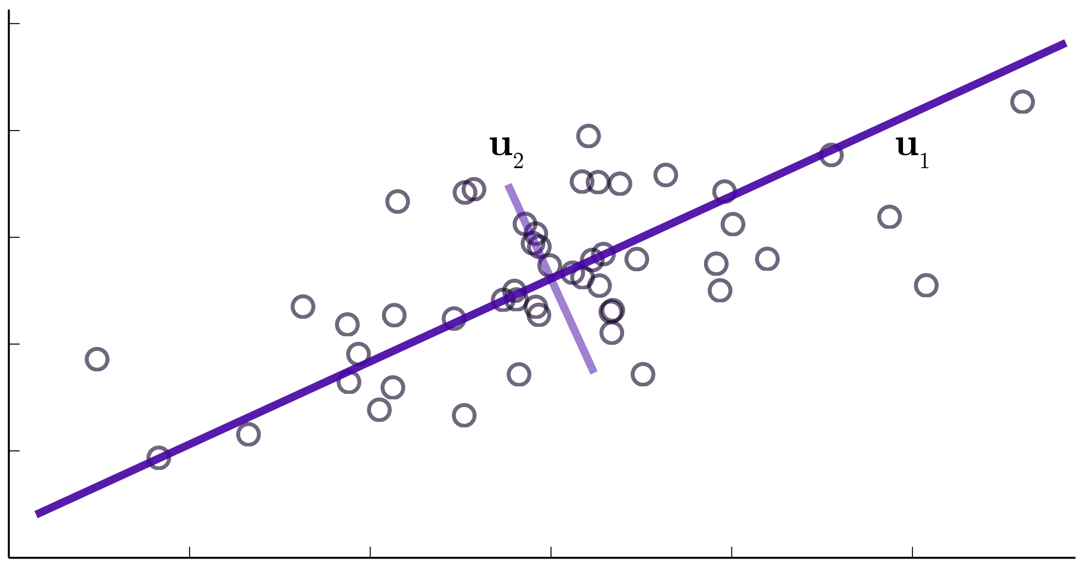


Figure 3.10: The first and second principle components for a set of 2-dimensional sample points.

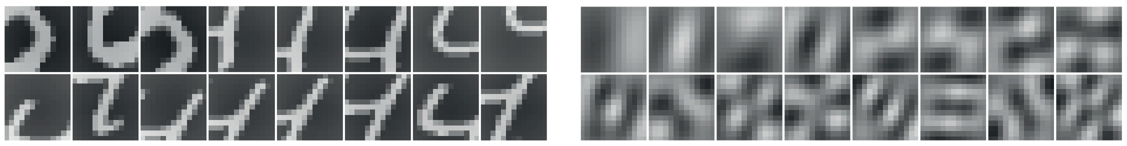


Figure 3.11: Examples of random 13x13 image patches after processing (left) and a subset of PCA basis functions (right).

3.2.2 Feature extraction

The coefficient images are then computed by subsequently filtering the image with basis images, i.e. the reshaped basis vectors. As a result, for each of the $K = 169$ basis vectors, a coefficient image $F_i(x, y)$ is produced. To achieve local shift invariance the extrema in non-overlapping regions are extracted by dividing every coefficient image $F_i(x, y)$ into 9 distinct 9x9 blocks. The last row and column are dropped as proposed by Labusch et al. [LBM08]. This means, for every coefficient image $F_i(x, y)$ a 9-dimensional vector $\boldsymbol{F}_i^+$ and $\boldsymbol{F}_i^-$ for the minima and maxima, respectively, is computed. The feature extraction process is outlined in Figure 3.12 where an exemplary digit images is filtered with the PCA basis functions and the maxima and minima are subsequently extracted in distinct 9x9 blocks.

The final feature vector $\boldsymbol{F}_I$ used for classification is then composed out of all $i = 1 \ldots K$ vectors $\boldsymbol{F}_i^+$ and $\boldsymbol{F}_i^-$:

$$\boldsymbol{F}_I = (\boldsymbol{F}_1^+, \boldsymbol{F}_2^+, \ldots, \boldsymbol{F}_K^+, \boldsymbol{F}_1^-, \boldsymbol{F}_2^-, \ldots, \boldsymbol{F}_K^-) \tag{3.21}$$

This resulting dimension D_I of the feature vector $\boldsymbol{F}_I$ is:

$$D_I = 169 \times 9 \times 2 = 3042 \tag{3.22}$$

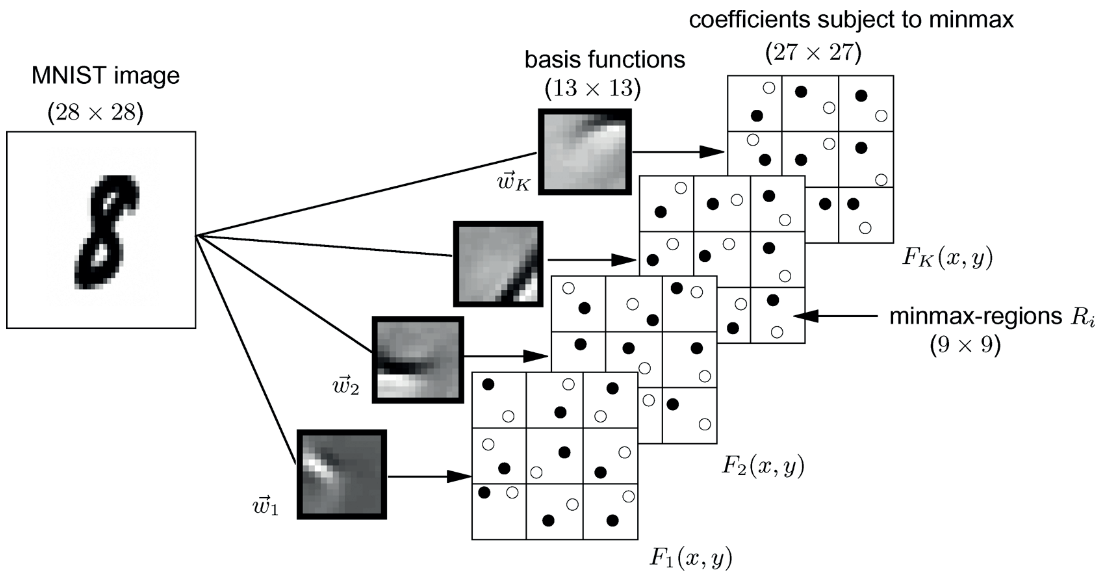


Figure 3.12: Schematics of the feature extraction process. The digit images (left) are filtered with the PCA basis functions in the filterbank (center) to compute the coefficient images (right). The combined extrema in distinct blocks are then used as the final feature vector [LBM08].

with $K = 169$ filters (basis functions), 9 distinct blocks and 2 extrema (maximum and minimum).

3.2.3 Classification

In the classification stage the class labels y_i for all feature vectors $\boldsymbol{x}_i$ produced in the previous steps are predicted. As proposed by Labusch et al. [LBM08] an SVM classifier with RBF kernel is used. Further, it is extended by the *one-versus-one* approach to support multiple classes and prediction values. The SVM is based on a *supervised learning* approach. The three main components of this approach are: randomly generated vectors $\boldsymbol{x}_i$, a supervisor and a learning machine [Bur98].

As shown in Figure 3.13, the supervisor acts as a black-box with an unknown underlying model and maps the input vectors $\boldsymbol{x}_i$ from a distribution $P(\boldsymbol{x})$ to the output labels y_i according to a conditional distribution function $P(y \mid \boldsymbol{x})$ [Vap99]. Both $P(\boldsymbol{x})$ and $P(y \mid \boldsymbol{x})$ are fixed but unknown. The task of the learning machine is to learn the underlying model, i.e. the mapping $\boldsymbol{x}_i \longmapsto y_i$. The supervisor is represented by a training set which is composed of $i = 1 \dots N$ independently drawn and identically distributed tuples $(\boldsymbol{x}_i, y_i)$ from a fixed distribution $P(\boldsymbol{x}, y) = P(\boldsymbol{x})P(\boldsymbol{x} \mid y)$ [Vap99]. Unfortunately in case of handwritten digit recognition, the unknown underlying model is rather complex as the supervisor is an actual person with subjective perception annotating images of handwritten digits.

The learning machine itself is defined as a set of functions $f(\boldsymbol{x}, \alpha)$, labeled by their parameters α, mapping $\boldsymbol{x} \longmapsto f(\boldsymbol{x}, \alpha)$ [Bur98]. The model is learned by selecting the function $f(\boldsymbol{x}, \alpha)$ which best describes the outputs y_i for inputs $\boldsymbol{x}_i$. This is done by minimizing a loss function as for instance the summed difference of the supervisor-outputs

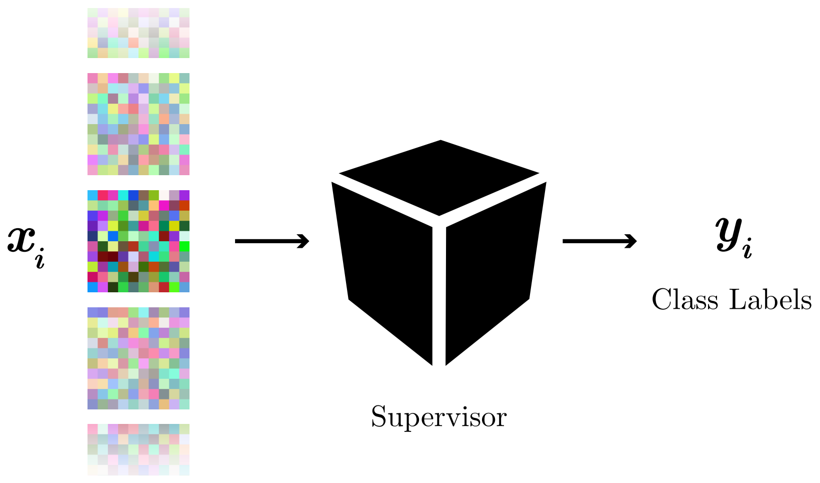


Figure 3.13: The supervised learning model: randomly generated input vectors $\boldsymbol{x}_i$ are handed over to a supervisor, i.e. black box, which subsequently outputs ground-truth labels y_i used to train a learning machine.

y_i and the predicted outputs $f(\boldsymbol{x_i}, \alpha)$, i.e. the mean error rate $R(\alpha)$ on the training set [Bur98]:

$$R(\alpha) = \sum_{i=1}^{N} |y_i - f(\boldsymbol{x}_i, \alpha)| \tag{3.23}$$

Support Vector Machines

The support vector machine classifier is a binary classifier. The decision boundary can be described using a linear function $y(\boldsymbol{x})$ of the input vector $\boldsymbol{x}$ [Bis06]:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + b \tag{3.24}$$

with weights $\boldsymbol{w}$ and bias b. Each input vector $\boldsymbol{x}_i \in \mathbb{R}^D$ is assigned a class label y_i according to the decision boundary:

$$y_i = \begin{cases} C_1 & y(\boldsymbol{x}) \geqslant 0 \\ C_2 & \text{otherwise} \end{cases} \tag{3.25}$$

Geometrically, this corresponds to separating D-dimensional data by a hyperplane of dimension $D-1$ defined as $y(\boldsymbol{x}) = 0$ with $\boldsymbol{w}$ denoting the normal vector of the hyperplane. The bias b controls the distance of the hyperplane to the origin of the coordinate system. Assuming an arbitrary point $\boldsymbol{x}$ and a point $\boldsymbol{p}$ on the hyperplane the shortest distance d of $\boldsymbol{x}$ to the hyperplane is defined as:

$$d = \frac{\boldsymbol{w}^T(\boldsymbol{x} - \boldsymbol{p})}{\|\boldsymbol{w}\|} = \frac{\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}^T\boldsymbol{p}}{\|\boldsymbol{w}\|} \tag{3.26}$$

with $\boldsymbol{x} - \boldsymbol{p}$ denoting the vector from $\boldsymbol{p}$ to $\boldsymbol{x}$. That means, the distance d is the length of the orthogonal projection of the vector $\boldsymbol{x} - \boldsymbol{p}$ onto $\boldsymbol{w}$. Since $\boldsymbol{p}$ is on the hyperplane $y(\boldsymbol{x}) = 0$, Equation 3.24 can be reformulated as

$$\boldsymbol{w}^T\boldsymbol{p} = -b \tag{3.27}$$

which shows that the distance corresponds to the normalized linear function $y(\boldsymbol{x})$ of Equation 3.24:

$$d = \frac{\boldsymbol{w}^T\boldsymbol{x} + b}{\|\boldsymbol{w}\|} = \frac{y(\boldsymbol{x})}{\|\boldsymbol{w}\|} \tag{3.28}$$

Since the hyperplane is oriented, i.e. *left* and *right* is defined by the direction of the normal $\boldsymbol{w}$, the side of the hyperplane the point $\boldsymbol{x}$ lies is reflected by the sign of d – which is the sign of $y(\boldsymbol{x})$. A hyperplane in $\mathbb{R}^2$ and the geometric explanation of calculating the distance of a point to this hyperplane is depicted in Figure 3.14.

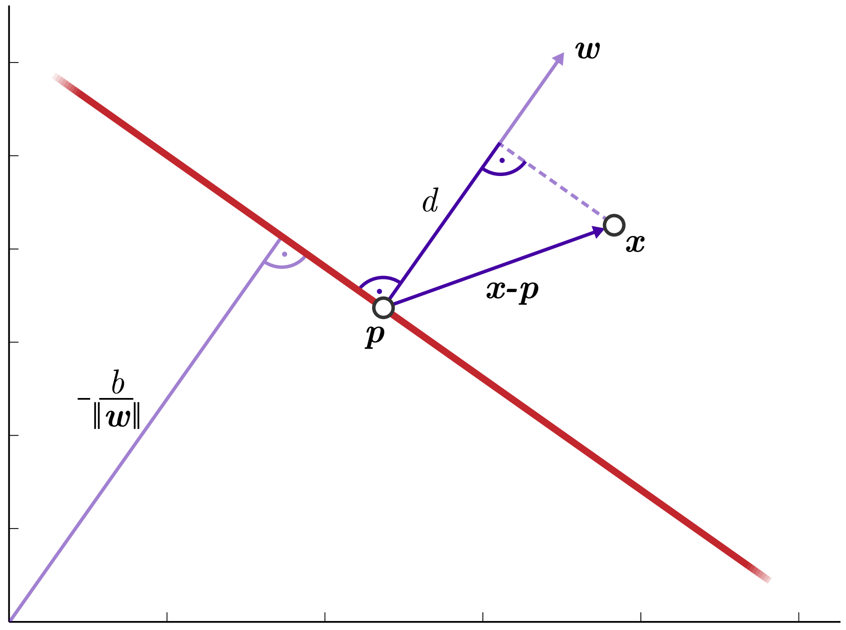

$\boldsymbol{w}$
d
$\boldsymbol{x}$
$\boldsymbol{x}$-$\boldsymbol{p}$
$\boldsymbol{p}$
$-\frac{b}{\|\boldsymbol{w}\|}$

Figure 3.14: Distance to a hyperplane (red line) explained in 2-dimensional space with normal vector $\boldsymbol{w}$ and bias b. The distance of point $\boldsymbol{x}$ to the plane is denoted with d. The point $\boldsymbol{p}$ is an arbitrary point on the hyperplane. However, this does not yet reflect the hyperplane of a trained SVM.

In case of linearly separable data there are infinitely many different solutions for this hyperplane [Bis06]. However, in SVMs the hyperplane is chosen which maximizes the margin, i.e. the distance to the closest point to the hyperplane. Assuming a training set with sample vectors $\boldsymbol{x}_i$ and given class labels $c_i \in \{-1, 1\}$ the set is linearly separable if a hyperplane $y(\boldsymbol{x}) = 0$ exists so that

$$c_i y(\boldsymbol{x}_i) > 0, \qquad i = 1 \ldots N \tag{3.29}$$

Using the definition of the distance to the hyperplane of Equation 3.28 the maximum margin hyperplane is found by solving [Bis06]:

$$\underset{\boldsymbol{w},b}{\arg\max} \left\{ \frac{1}{||w||} \min_{i=1\ldots N} \left[c_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b \right) \right] \right\} \tag{3.30}$$

By rescaling $\boldsymbol{w}$ and b with a factor κ the distance to the hyperplane

$$d = \frac{\kappa \hat{\boldsymbol{w}}^T \boldsymbol{x} + \kappa \hat{b}}{\|\kappa \hat{\boldsymbol{w}}\|} = \frac{\boldsymbol{w}^T \boldsymbol{x} + b}{\|\boldsymbol{w}\|} \tag{3.31}$$

remains unchanged, however it allows setting

$$c_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b \right) = 1 \tag{3.32}$$

for the point $\boldsymbol{x}_i$ closest to the hyperplane. Since this point is closest to the hyperplane, the distance to the hyperplane must be greater or equal for all points $\boldsymbol{x}_i$, so that

$$c_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b \right) \geqslant 1, \qquad i = 1 \ldots N \tag{3.33}$$

Therefore solving Equation 3.30 simplifies to maximizing $1/||\boldsymbol{w}||$ due to Equation 3.32 and 3.33

$$\min_{i=1 \ldots N} \left[c_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b \right) \right] = 1 \tag{3.34}$$

Further, the maximization problem can be reformulated as finding parameters $\boldsymbol{w}$ and b which minimize $||\boldsymbol{w}||^2$ under the constraints of Equation 3.33 [Bis06]. For further convenience, the Lagrangian dual problem is formulated. The first step for obtaining the dual problem is introducing Lagrange multipliers α_i for the constraints [Bis06]. The Lagrangian function for the optimization problem is then given by:

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}||\boldsymbol{w}||^2 - \sum_{i=1}^{N} \alpha_i \left[c_i(\boldsymbol{w}^T \boldsymbol{x}_n + b) - 1 \right] \tag{3.35}$$

The factor $1/2$ does not change the location of the maximum and is used for later convenience. The solution to the problem is obtained by setting the derivatives of the Lagrangian function to zero:

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{i=1}^{N} \alpha_i c_i \boldsymbol{x}_i = 0 \qquad \frac{\partial L}{\partial b} = \sum_{i=1}^{N} \alpha_i c_i = 0 \tag{3.36}$$

By substitution of $\boldsymbol{w}$ in the Lagrangian function in Equation 3.35,

$$L_D(\boldsymbol{a}) = \frac{1}{2} \left(\sum_{i=1}^{N} \alpha_i c_i \boldsymbol{x}_i \right)^2 - \sum_{i=1}^{N} \alpha_i c_i \left(\sum_{j=1}^{N} \alpha_j c_j \boldsymbol{x}_j \right) \boldsymbol{x}_i - b \underbrace{\sum_{i=1}^{N} \alpha_i c_i}_{=0, \text{ due to (3.36)}} + \sum_{i=1}^{N} \alpha_i \tag{3.37}$$

$\boldsymbol{w}$ and b are eliminated and the dual problem is obtained:

$$L_D(\boldsymbol{a}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j c_i c_j (\boldsymbol{x}_i \boldsymbol{x}_j) \tag{3.38}$$

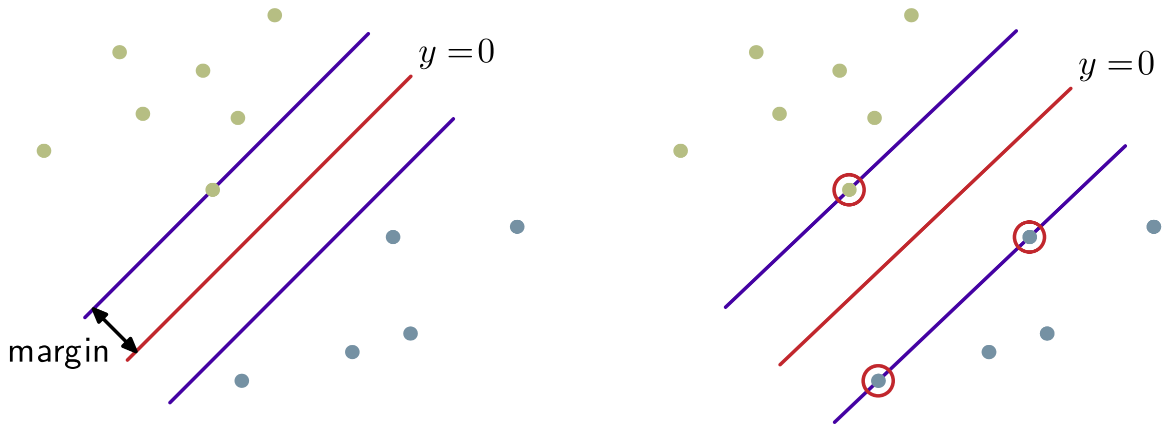


Figure 3.15: In SVMs the decision boundary (red) is optimized for a maximal margin (purple). Even though the decision boundaries left and right both separate the sample vectors, the boundary on the right is chosen by the SVM as it maximizes the margin. The sample vectors closest to the decision boundary, i.e. on the margin, are called *support vectors* (encircled) [Bis06].

In addition to the general constraint of Lagrangian multipliers $\alpha_i \geqslant 0$, the second constraint of Equation 3.36 still holds:

$$\sum_{i=1}^{N} \alpha_i c_i = 0 \tag{3.39}$$

The duality gap, i.e. the gap between the solution of the primal problem and the dual problem is zero because the primal problem is convex [Bis06]. The parameters α_i can be found using a *Quadratic Programming* algorithm [Bis06]. Subsequently, the parameter b is obtained by substituting the computed α_i in Equation 3.32.

Returning to the linear decision function in Equation 3.25 and substituting $\boldsymbol{w}$ according to Equation 3.36, the classification of new sample vectors $\boldsymbol{x}$ is achieved using

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i c_i (\boldsymbol{x}_i \boldsymbol{x}) + b \tag{3.40}$$

with the trained parameters α_i and b, and the training samples $\boldsymbol{x}_i$ with corresponding class labels c_i. The training samples $\boldsymbol{x}_k$ with $\alpha_k = 0$ are not contributing to the prediction. However the remaining sample vectors, i.e. the vectors lying on the margin to the hyperplane, are called *support vectors*. In Figure 3.15 the decision boundary and the support vectors are depicted on an example.

Kernel-Trick

The main advantage of the dual representation lies in the occurrence of the sample vectors $\boldsymbol{x}$. Assuming a transformation $\boldsymbol{x} \mapsto \phi(\boldsymbol{x})$ from an input space $\mathbb{R}^D$ to a feature space $\mathbb{R}^M$

and substituting $\boldsymbol{x}$ and $\boldsymbol{x}_i$ in the linear decision function of Equation 3.40

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i c_i (\phi(\boldsymbol{x_i})\phi(\boldsymbol{x})) + b \tag{3.41}$$

shows, that the single occurrence of the mapping ϕ is in the inner product of the vectors $\phi(\boldsymbol{x_i})$ and $\phi(\boldsymbol{x})$. By defining a so called *kernel function* or *kernel* $\phi(\boldsymbol{x_i}, \boldsymbol{x})$ the explicit mapping into feature space by computing $\phi(\boldsymbol{x_i})$ and $\phi(\boldsymbol{x})$ is omitted [Bis06]. Instead, the inner product in feature space is given by the kernel. The decision function can then be written as

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i c_i \phi(\boldsymbol{x_i}, \boldsymbol{x}) + b \tag{3.42}$$

The simplest kernel is the linear kernel which is defined as the inner product of the vectors $\boldsymbol{x_i}$ and $\boldsymbol{x}$ [CKLS07]. Other popular kernels are the polynomial kernel, the sigmoid kernel and the RBF kernel [CKLS07] which is used for the proposed classification method:

$$\phi(\boldsymbol{x_i}, \boldsymbol{x}) = \exp\left(\frac{-||\boldsymbol{x} - \boldsymbol{x}_i||^2}{2\sigma^2}\right) \tag{3.43}$$

The advantage of non-linear kernels are that they allow the separation of linearly non-separable sample vectors. This is also called the *kernel trick* and is done by mapping the input vectors into a higher dimensional feature space in which a separating hyperplane (with maximal margin) can be found. For instance, the two dimensional input samples in Figure 3.16 (a) are not linearly separable. However, after mapping them onto a three dimensional parabola (see (b)), a separating hyperplane can be found (see (c)).

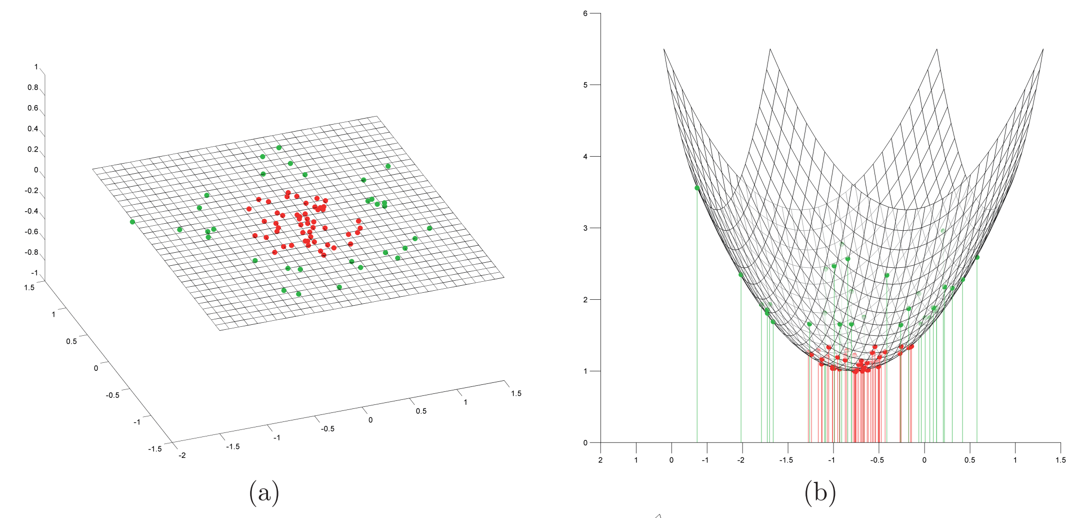
There are two reasons an RBF kernel is chosen for the classification of the feature vectors of the handwritten digits. Firstly, an SVM with a linear kernel can be replicated using an SVM with an RBF kernel with $\sigma \to \infty$ [KL03]. Therefore, in case the whole (C, σ) parameter space is searched, an SVM with RBF kernel is at least as good as an SVM with a linear kernel. However, as previously stated the RBF kernel additionally allows a separation of linearly non-separable data. Secondly, the polynomial kernel has the disadvantage that it has more hyperparameters and therefore the search for the optimal parameters is more complex and hence computationally expensive.

Soft-Margin

Although the use of a kernel allows the transformation of the data into a higher dimensional feature space, only cases in which the data is linearly separable in this space have been accounted so far. This is especially problematic as there is no guaranteed linear separability for real world problems. To overcome this problem, a slack variable ξ_i is introduced to allow missclassified training samples [Bis06]. This is done by alleviating the constraint of Equation 3.33:

$$c_i \left(\boldsymbol{w}^T \boldsymbol{x}_i + b\right) \geqslant 1 - \xi_i, \quad \xi_i \geqslant 0, \, i = 1 \dots N \tag{3.44}$$

This way, not all sample vectors must be outside the margin - therefore the term *soft-margin.* However the slack variable acts as a penalty [Bis06]. For samples inside the


(a)

(b)

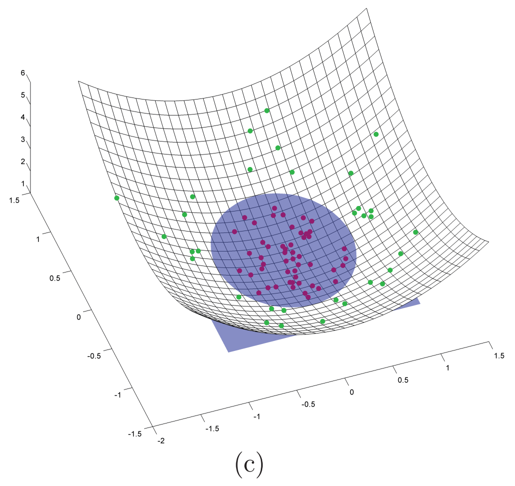

(c)

Figure 3.16: Kernel trick visualized using two dimensional feature vectors and a polynomial kernel. The two classes (red and green) in (a) are not linearly separable in two dimensions. However, after mapping the points onto a three dimensional parabola using a polynomial kernel (b) the classes can be separated linearly using a plane (c).

margin $0 < \xi_n < 1$ for correctly classified samples, i.e. samples on the correct side of the hyperplane, and $\xi_i > 1$ otherwise. For points on the margin or on the correct side outside the margin $\xi_i = 0$. Therefore, instead of minimizing $||w||^2$ the optimization problem is reformulated as minimizing [Bis06]:

$$C\sum_{i=1}^{N}\xi_i + \frac{1}{2}||w||^2 \tag{3.45}$$

The parameter C controls the influence of the slack variable, i.e. the penalty for outliers. As shown above this problem can again be reformulated using Lagrangian multipliers and the dual problem is similar to the one described in Equation 3.38. However, the constraint for the coefficients α_i is tightened as C is added as an upper bound [Bis06]:

$$0 \leqslant \alpha_i \leqslant C \tag{3.46}$$

Points inside the margin have $\alpha_i = C$ and for points outside the margin $0 < \alpha_i < C$ [Bis06]. Since those points have $\xi_i = 0$, Equation 3.32 can again be used to compute b.

There is no guarantee for the separability of real world data which also applies to the classification of handwritten digits. Furthermore, residual errors in the ground truth and outliers may produce feature vectors which are similar to digits of another class. Therefore, the use of a soft-margin is crucial for the digit recognition as these spurious samples would otherwise form undesired islands in the decision boundary which decrease the generalization ability of the SVM.

SVM Hyperparameter Estimation

For the use of SVMs with RBF kernel and softmargin two hyperparameters, namely C and γ have to be estimated. As proposed by Chang et al. [CL11] this is achieved using cross-validation with a logarithmic grid for the hyperparameters. For the cross validation the training set is separated into $k = 3$ distinct subsets. In each iteration, one subset is selected as the test set and the other subsets are used as the training set. The results of all k iterations are then averaged. The cross-validation is performed for each (C, γ) position on the grid and the hyperparameters that maximize the cross-validation accuracy are chosen.

However, this approach is computationally expensive, since for each position on the grid k iterations of training and prediction have to be performed. Therefore, in a first step a coarse grid is chosen and after manually inspection a finer grid is used to estimate the final hyperparameters. In Figure 3.17 the cross-validation results for the CVL-HDRC (see Section 4.1) training set on grid of $\gamma = 2^{-9}, 2^{-8}, \dots 2^{-1}$ and $C = 2^{-1}, 2, \dots 2^{9}$ are visualized.

Multiclass Support

As described above the SVM is a binary classifier. However, in case of digit recognition multiple $K = 10$ classes have to be classified. There are two popular approaches for solving this problem [Bis06]. On the one hand, in the *one-versus-the-rest* approach K SVMs are constructed, i.e. one for each class. Each SVM is trained to separate one of the classes with all other classes. On the other hand, the *one-versus-one* approach uses

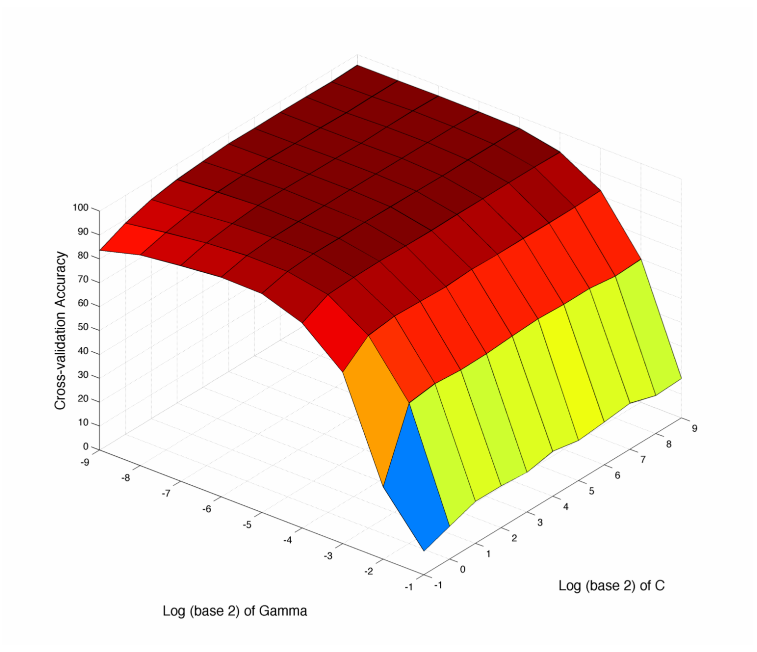


Figure 3.17: Cross-validation accuracy for the CVL-HDRC training set over the hyper-parameters γ and C.

$K(K-1)/2$ different SVMs with each SVM separating one class from another. The class label is then assigned using a voting scheme, i.e. the class with the most votes is selected. An advantage of the latter approach is, that pairwise probabilities can be estimated in a succeeding step. However, when dealing with small numbers of training samples the former approach is beneficial.

Yet, as a high number of training samples is available for the digit recognition and additionally, the estimation of class probabilities is desired, the one-against-one approach is advantageous for the system proposed.

Prediction Values

In order to allow the meaningful rejection of uncertain predicted class labels the certainty that an input vector $\boldsymbol{x}$ belongs to a class $i \in \{1, \ldots, k\}$ has to be estimated, i.e. the probabilities

$$p_i = P(c = i \,|\, \boldsymbol{x}), \qquad i = 1, \ldots, k \tag{3.47}$$

However, the output of an SVM is a predicted class label without probability information. To overcome this problem, Chang et al. [CL11] propose a method to extend SVMs for estimating these probabilities. Assuming a classification problem with two classes, it is sufficient to estimate the pairwise probabilities r_{ij}

$$r_{ij} \approx P(c = i \,|\, c = i \,\mathrm{or}\, j,\, \boldsymbol{x}) \tag{3.48}$$

as there are only two classes (i and j) and therefore:

$$P(c = i \,|\, c = i \,\mathrm{or}\, j,\, \boldsymbol{x}) = P(c = i \,|\, \boldsymbol{x}) = 1 - P(c = j \,|\, \boldsymbol{x}) \tag{3.49}$$

Chang et al. [CL11] estimated these pairwise probabilities as

$$r_{ij} \approx \frac{1}{1 + e^{Ay+B}} \tag{3.50}$$

where y is the decision value for an input vector $\boldsymbol{x}$. As the training samples provide input vectors and labels the parameters A and B can be computed by minimizing a likelihood function [CL11]. Using the one-against-one approach for multiclass problems, the pairwise probabilities can be estimated the same way. However, Equation 3.49 does not hold in this case. Therefore, the equality

$$P(c = j \mid c = i \,\mathrm{or}\, j,\, \boldsymbol{x}) \cdot P(c = i \mid \boldsymbol{x}) = P(c = i \mid c = i \,\mathrm{or}\, j,\, \boldsymbol{x}) \cdot P(c = j \mid \boldsymbol{x}) \tag{3.51}$$

is used to formulate the following minimization problem:

$$\min_{\boldsymbol{p}} \left[\frac{1}{2} \sum_{i=1}^{k} \sum_{j:j\neq i} (r_{ij}p_i - r_{ij}p_j)^2 \right], \quad \text{with} \quad p_i \geqslant 0, \quad \sum_{i=1}^{k} p_i = 1 \tag{3.52}$$

which can again be solved using *Quadratic Programming* [CL11]. The computation of these prediction values provides the means to identify uncertain digits and either highlight them for manual inspection or reject them if a certain minimal recognition accuracy is required.

Summary

In this chapter the methods used for the extraction and recognition of the digits and signs in the weather records were described. Their localization is based on reconstructing the tabular structure of the forms using line detection. The use of projection profiles for this task is motivated by the fact that this approach is applicable for grayscale images and therefore a binarization is avoided. This provides the advantage that the form lines are emphasized as they are darker than the handwritten strokes. As this method searches for peaks in horizontal and vertical projection profiles, it depends on axis aligned images. Therefore, an approach for rotation correction using a direction histogram of the gradient vectors was presented. It allows a precise estimation of the main orientation due to an emphasis of pixels belonging to strokes and form lines and the suppression of non-significant maxima. For the detection of the maximum in the histogram a fitted polynomial is used.

Further, as spurious horizontal and vertical lines influence the recognition performance, a stroke preserving line removal method was proposed which removes the spurious lines while preserving the handwritten strokes by reconstructing gradient images using a Wiener filter. In contrast to other methods, this approach is not based on the reconstruction of binary images using heuristics, which discards the intensity differences present in grayscale images, but instead it exploits those differences.

Moreover, the feature extraction using a trained PCA filter bank was explained in detail. It is based on the biologic visual system and provides a trainable feature extractor. Additionally, shift-invariance is achieved by extracting maxima and minima in distinct image blocks. Further, an overview of SVMs was given. The construction of the linear decision boundary was described and the advantage of formulating the dual problem for the use of kernel functions was depicted. In contrast to NN, the SVMs provide a unique solution, i.e. a global maximum, as instead of any decision boundary separating the training points, the boundary with maximal margin is chosen. Furthermore, the construction of SVMs only requires the optimization of two hyperparameters (in case of an RBF kernel) instead of fine-tuning the number of neurons and layers for NN. An RBF kernel was chosen for the SVMs, since it provides at least the same performance as a linear kernel and has less hyperparameters as a polynomial kernel. Moreover, the use of a soft-margin was motivated by the fact, that real-world data can not be assumed to be linearly separable and the decision boundary would otherwise form undesired islands which reduces the ability to generalize.

Additionally, since it is desired to identify uncertain classification results, the computation of the prediction values, i.e. the estimated class probabilities, was outlined. Further, it was depicted that the one-versus-one approach for supporting multiclass problems is advantageous over the one-versus-the-rest approach, since it allows the estimation of pairwise probabilities which is required for the computation of these prediction values.

Chapter 4

Results

In this chapter the system presented is empirically evaluated. To identify the strengths and weaknesses of individual parts of the system, multiple experiments are conducted. In case manually annotated data is available the evaluation is based on this data. Otherwise, appropriate datasets are generated synthetically. The datasets can be divided into three categories: firstly, three different digits databases with manually annotated ground truth. Secondly, synthetically generated data for the evaluation of the line removal approach and thirdly, weather records with manually transcribed temperature values.

In the first section of this chapter the digit recognition method is evaluated using three different databases namely MNIST, USPS and CVL. The differences regarding locality, writers, preprocessing, etc. between these databases are elaborated. Further, experiments are conducted to evaluate the accuracy of the digit recognition system and to show its strengths and weaknesses for the different digit classes. Further, an experiment is conducted to evaluate the validity of the prediction values. Additionally, the human error is quantified by examining the results of a manual annotation process.

Secondly, in Section 4.2, the line removal method proposed is investigated using synthetically generated data in order to detect cases where this method is applicable and where its limitations are. Finally, in Section 4.3 the whole system starting from scanned images to classified digit and sign cells is evaluated. This way, in addition to the digit and sign recognition accuracy, strengths and weaknesses of the segmentation process are shown. The experiments are conducted using two datasets of weather records from five different measuring stations in Austria.

4.1 Digit Recognition

In this section, the performance of the digit recognition module is evaluated using datasets containing manually annotated handwritten digits. The different databases vary in size, number of writers, preprocessing steps performed beforehand, scanning- and final image resolution, etc. Further, the MNIST and USPS databases were collected among North American writers, whereas our own database – from now on referenced as CVL database – was collected among European writers whose writing styles differ significantly for some digits like one and seven.

The performance of the digit recognition module is on the one hand evaluated by computing the prediction accuracies achieved. This is carried out using the given trainings sets for training and predicting the class labels of the digits in the test sets. On the other hand, the strengths and weaknesses of the proposed method for distinguishing certain pairs of digits are elaborated. Additionally, since the CVL database allows an identification of each digit's writer, experiments are conducted to identify the systems robustness with respect to different writing styles, i.e. digits from unseen writers.

The accuracies achieved on these databases serve as an upper bound for the expected accuracy on the weather records as these digits were either manually or semi-manually segmented beforehand. That means, there is no bias due to failures in the segmentation process.

4.1.1 Databases

In this section the different handwritten digit databases, namely the MNIST, USPS and CLV, are described in detail and there differences are investigated.

MNIST

The MNIST database [LC] was constructed using the Special Database (SD-1) and Special Database 3 (SD-3) from the United States National Institute of Standards and Technology (NIST) [Gro95]. The digits in the SD-1 were collected among high-school students, whereas the digits in the SD-3 among Census Bureau employees. Due to this fact, the digits in the SD-3 are clearer than in SD-1. The digits in SD-1 and SD-3 were scanned at 300dpi with each digit stored in a 128x128 binary image. The ground truth labels were manually checked with an approximate remaining error of about 0.1% [Gro95]. Originally, the SD-1 was used as a test set and SD-3 as a training set. However, due to the differences in quality of the digit images and the fact that for supervised learning the samples in the training set must be identically and independently drawn form the whole set [Vap99], the MNIST database was created by mixing SD-1 and SD-3.

For the MNIST database, 60,000 digits were selected equally from SD-1 and SD-3 for the training set and 10,000 for the test set. The writers in those sets are disjoint and the training set contains digits from about 250 writers. The digits were size normalized to 20x20 pixels and the centroid was placed in the center of a 28x28 pixel reference frame. Due to the normalization process and the inherent anti-aliasing the final images are grayscale (8-bit). In Figure 4.1 a random selection of digits from the MNIST database is shown.

USPS

The USPS database was created by the Center of Excellence in Document Analysis and Recognition (CEDAR) at the State University of New York at Buffalo using mail from the US Postal Service [Hul94]. The data was scanned at 300dpi and each digit is available as a normalized 16x16 pixel grayscale image. The training set contains 7,291 digits, whereas the test set contains 2,007 digits. Figure 4.2 shows a random selection of digits from the USPS database.

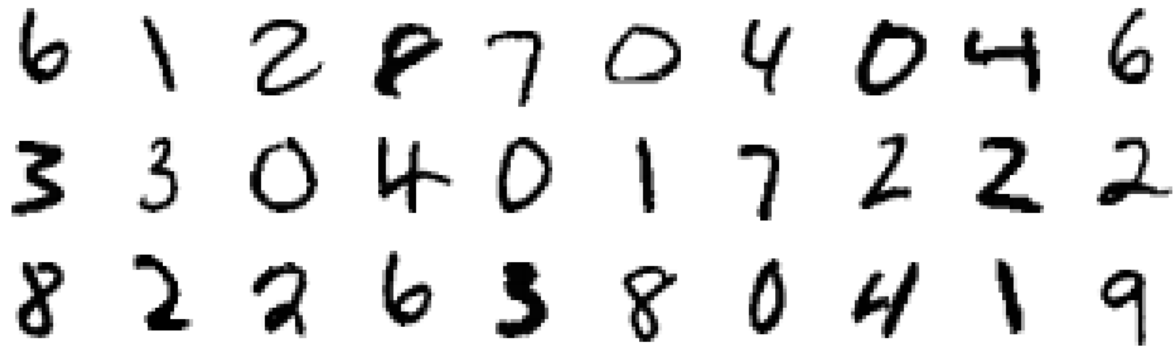


Figure 4.1: Random selection of digits from the MNIST database.

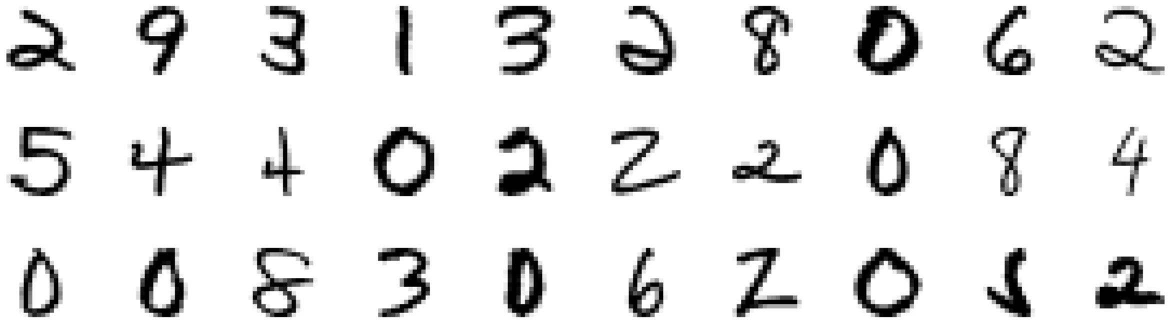


Figure 4.2: Random selection of digits from the USPS database.

CVL

The CVL digit database was created by the Computer Vision Lab at the Vienna University of Technology. It was collected among 303 secondary school and university students, male and female alike. Similarly to the MNIST and the USPS database the digits were scanned at 300dpi. However, in contrast to other digit databases available the digit images are provided un-normalized in RGB. The digits were extracted from 26 different digit strings. Further, the ground truth labels were generated manually and at least reviewed once. As this database is closest to the real world data from all presented databases it is also the most challenging. The high variance in writing styles can be observed in 30 randomly selected digit images from this database shown in Figure 4.3. Another unique feature of this database is that the writers are from European origin which can for instance be noticed in the digits '1' and '7'.

The whole set considered here consists of about 35,000 digit images. The following subsets of the database were used for evaluation:

CVL-HDRC: a training, validation and test set publicly available and used for the Handwritten Digit Recognition Competition (HDRC) at the ICDAR 2013. The training and validation sets consist of 7,000 digits each, which are evenly distributed over all 10 classes, i.e. 700 per class. The test set is made up of 21,780 images, with 2,178 digits per class. The writers from all sets are disjoint.

CVL-Random: is generated from a superset of the CVL-HDRC training and validation set containing about 17,000 samples; 400 random samples from each class were selected as the test set. This accumulates to a test set size of 4,000 samples. The remaining samples were used for training. The writers are not disjoint in these sets.

CVL-Small: 2,000 digits randomly chosen (200 per digit) from the CVL-Random test set.

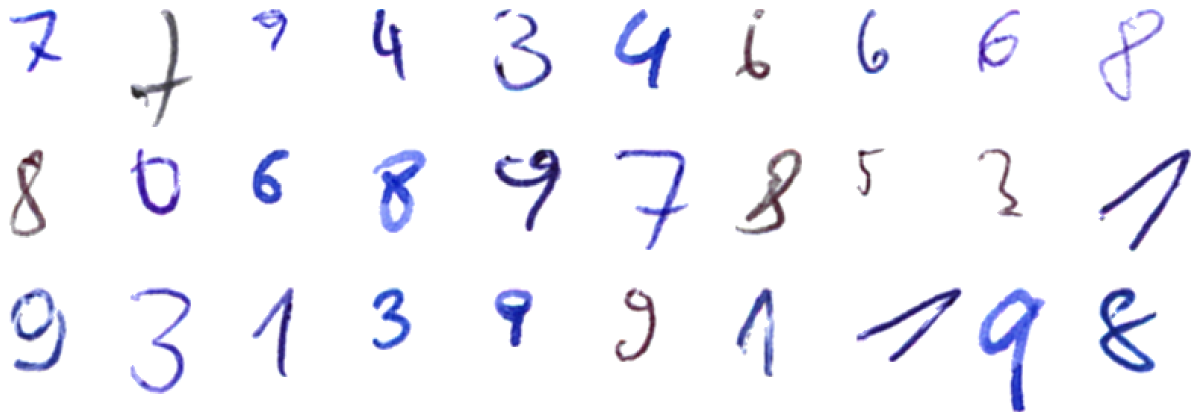


Figure 4.3: Random selection of digits from the CVL database.

4.1.2 Classification Accuracy

In the training phase, first the metaparameter for the RBF-SVMs (γ and C) are trained using 3-fold cross-validation as described in Section 3.2.3. For the computation of the PCA basis, 1,000 random patches are used from the training set. The SVMs are then subsequently trained using the parameters with the best cross-validation accuracy. To identify the raw prediction performance, in a first step the recognition accuracy is calculated for each database. If not specified otherwise, the training set of the database was used for training and the test set for evaluation. The accuracy is computed as:

$$\text{accuracy} = \frac{\text{number of correctly classified samples}}{\text{number of samples}} \tag{4.1}$$

The results in Figure 4.4 show, that the recognition accuracy for the proposed methods lies between 93.27% and 99.24%. The best result is achieved on the MNIST database and the worst result on the CVL-HDRC validation set. Further, the results support that the USPS database is more challenging than the MNIST database. Due to the inferior image quality an accuracy of only 97.29% is achieved on this database. By interchanging the training and test set the accuracy drops about 1-2%. The larger drop of 2% for the CVL-Random dataset is reasonable, since on the one hand, the training dataset is fairly small (200 samples per class) and on the other hand, the CVL-Random dataset is overall more challenging than MNIST. The difference of about 3% between CVL-Random and CVL-HDRC and CVL-HDRC Valid shows that separating the writers in the training and test set increases the difficulty of the problem.

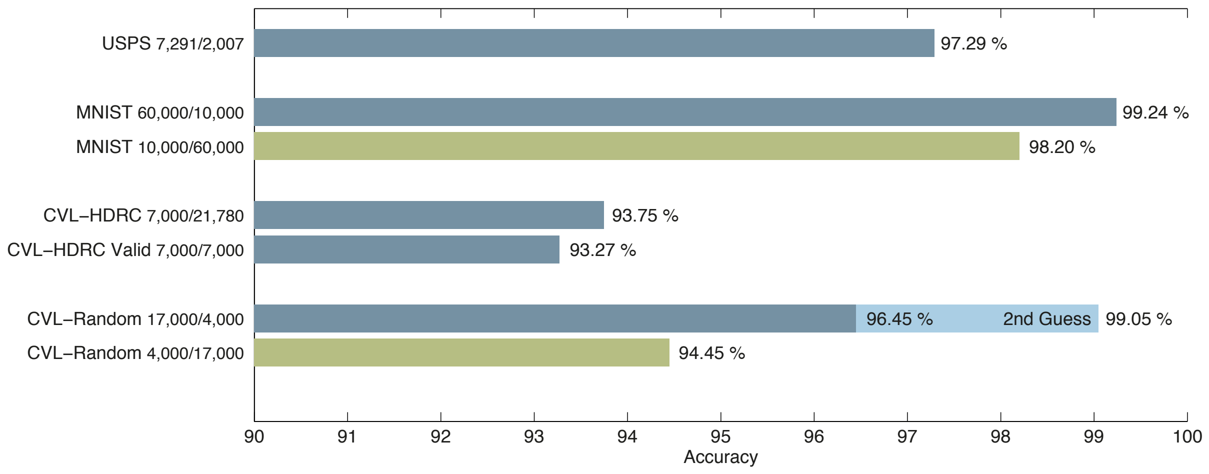


Figure 4.4: Digit recognition accuracies for various databases. The numbers indicate the size of the training and test set, respectively.

For the computation of the second guess two classes are predicted and the sample is classified correctly if one of the classes predicted is correct. This is done using the prediction values computed and improves the performance by 2.5% on the CVL-Random dataset. As the accuracy is improved compared to a randomly generated second guess (96.84%) by over 2%, this indicates that the prediction values provide meaningful results. However, this will be evaluated in Section 4.1.5 in detail. Generally, it has to be noted that the CVL database is the most challenging as it is closest to real world data of all databases.

Since the MNIST database is publicly available it allows a comparison of the performance of the proposed method to state-of-the-art digit recognition systems. The accuracy of the proposed method is similar to the accuracy of 99.20% achieved by Labusch et al. [LBM08] on whom it is based on. However, they additionally propose the use of a sparse basis functions which outperform the method proposed with an achieved accuracy of 99.41%. The CNNs proposed by Lecun et al. [LBBH98] perform slightly worse with 99.05%, although improved versions of this approach with more complex NN and a synthetically increased training set outperform all methods with an accuracy of 99.87% [CMS12]. Other approaches presented in Section 2.1.3 and 2.2.2 achieve accuracies between 99.0% and 99.5% on the MNIST database.

For the USPS database Keysers et al. [KDGN07] achieve a similar accuracy of 97.5% as the proposed method with 97.3%.

4.1.3 Challenges

In the previous section the overall performance of the proposed method was evaluated. However, to provide a more thorough evaluation of the specific challenges for digit recognition, additional metrics are defined and it is identified which classes are frequently confused. First of all, the multiclass classification problem is divided into ten different binary sub-problems - one for each class. Assuming a fixed class C, a sample of class

C_i with predicted class C_p it is either in class C or not. Subsequently, true positives are defined as samples where $C_i = C$ and $C_p = C$. Further, false positives are defined as samples where $C_i \neq C$ but $C_p = C$ and false negatives are samples where $C_i = C$ but $C_p \neq C$. Using the number of false positives, allows to identify which C_i are frequently misclassified as C. This is shown in Table 4.1 for the CVL-HDRC test set.

		predicted class										
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	$\sum$
true class	**0**	**96.7**	·	·	·	·	·	1.2	·	1.5	·	100.0
	1	·	**99.2**	·	·	·	·	·	·	·	·	100.0
	2	0.6	1.0	**94.7**	0.6	·	·	·	1.3	1.3	·	100.0
	3	·	0.9	1.3	**90.4**	·	1.8	·	1.1	1.6	2.5	100.0
	4	·	0.9	·	·	**95.3**	·	0.8	0.6	0.5	1.7	100.0
	5	·	·	·	1.9	0.7	**94.7**	·	·	1.1	0.7	100.0
	6	1.1	·	·	·	0.7	0.6	**95.4**	·	1.3	·	100.0
	7	·	0.5	·	0.5	2.4	·	·	**93.6**	0.8	1.7	100.0
	8	·	0.7	0.9	0.8	·	0.9	0.8	·	**93.9**	1.3	100.0
	9	0.6	2.6	·	2.6	1.9	1.1	·	0.8	6.6	**83.7**	100.0
	$\sum$	99.6	106.2	98.2	97.0	101.8	99.5	98.6	98.1	109.0	92.0	**93.8**

Table 4.1: Confusion matrix for the digit recognition on the CVL-HDRC test set. The rows indicate the actual class label extracted from the ground truth, whereas the columns indicate the prediction results. All value are given as fractions (in %) of the whole number of digit images per digit class, i.e. 2178. Values below 0.5% have been omitted for clarity. The diagonal represents the number of true positives with the average in the last row.

The values in the diagonal represent the true positives, whereas the other values depict the false positives. This shows for instance that the digit '9' is often (6.6%) misclassified as '8'. Additionally, the three highest numbers of false positives stem from the digit '9' which explains the low number of true positives of 83.7%. Further, all digits except '1' are more than 0.5% misclassified as '8' and even though the digits are evenly distributed among all classes the label '8' is assigned 1.09 times the number of samples in each class. The highest number of true positives is achieved for the digit '1', however its label is also assigned commonly, i.e. 1.06 times the average.

From the confusion matrix two metrics can be extracted. Firstly, the precision which is indicated in the diagonal is the ratio of the correctly classified samples of a class to the total number samples in this class:

$$\text{precision} = \frac{\text{number of true positives}}{\text{number of true positives + false positives}} \tag{4.2}$$

Secondly, the recall is defined as the ratio of the samples correctly classified as one particular class to the total number of all samples classified as that class. In Table 4.1 these are the diagonal values divided by the values in the last row, respectively for each column:

$$\text{recall} = \frac{\text{number of true positives}}{\text{number of true positives + false negatives}} \tag{4.3}$$

Using the harmonic mean of precision and recall a combined measure for the performance called the *F-score* can be derived:

$$\text{F-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{4.4}$$

The values of the F-score are in the interval [0,1] with 1 indicating the best performance and 0 the worst. A high F-score is only achieved if both precision and recall are high. In Figure 4.5 the F-score, precision and recall are depicted for the CVL-HDRC test set. The results support the previous observation that the performance is worst for the digits '8' and '9'. However, although the F-score is similarly low for these digits, the reasons are different. On the one hand, for the digit '8' this is due to a low precision, but on the other hand, for the digit '9' the reason is a low recall.

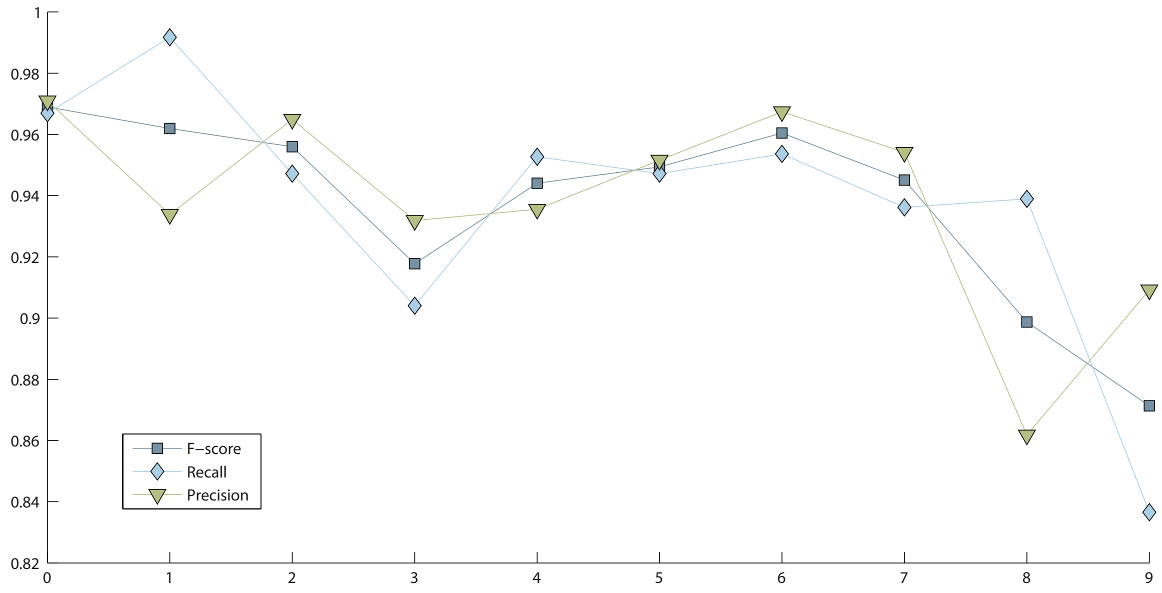


Figure 4.5: F-score, precision and recall for each digit class on the CVL-HDRC test set.

For comparison the confusion matrix (see Table 4.2) and F-score, precision and recall (see Figure 4.6) were additionally computed for the MNIST database. In this case the digit '7' performs worst due to a low precision. Especially the digits '2' is often (0.4%) confused as '7' and similarly, the digit '7' is often (0.5%) confused as '2'. This can be explained by the fact, that North American writers omit the horizontal stroke for the digit '7' and therefore it is more easily confused with the digit '2'.

Furthermore, in contrast to the CVL database the digits '8' and '9' achieve an average performance. This leads to the assumption, that for these digits the variation is particularly high in the CVL database. Although for the digit '1' the writing styles of North American writers differ from European writers, for both databases this digit achieves a high performance due to the highest recall of all classes and an average precision. However, the reasons for the low precision values differ. For the CLV database it is caused by a frequent misclassification of the digit '9', whereas for the MNIST database by a frequent misclassification of the digit '7'.

	predicted class										
true class	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	$\sum$
0	**99.8**	·	·	·	·	·	0.1	0.1	·	·	980
1	·	**99.8**	0.1	·	·	·	·	·	·	0.1	1135
2	0.2	·	**98.9**	·	0.2	·	·	0.4	0.2	0.1	1032
3	·	·	0.1	**99.0**	·	0.3	·	0.3	0.3	·	1010
4	·	·	·	·	**99.2**	·	0.5	·	·	0.3	982
5	0.1	0.1	·	0.4	·	**99.0**	0.1	0.1	0.1	·	892
6	0.4	0.1	·	·	0.1	0.1	**99.1**	·	0.2	·	958
7	·	0.3	0.5	0.1	·	·	·	**99.0**	·	0.1	1028
8	0.1	·	0.1	0.2	0.1	·	0.1	0.1	**99.1**	0.2	974
9	·	·	·	·	0.4	0.1	·	0.3	0.1	**99.1**	1009
10	986	1138	1029	1007	982	888	957	1031	974	1008	**10000**

Table 4.2: Confusion matrix for the digit recognition on the MNIST test set. The rows indicate the actual class label extracted from the ground truth, whereas the columns indicate the prediction results. All values except the last row and column are given as fractions (in %) of the whole number of digit images per class. The diagonal represents the number of true positives. The last row and column represent the number of samples in each column and row, respectively.

4.1.4 Human Error

In order to set the recognition accuracies in relation to the manual transcription of digits, the human error is quantified. Although that for instance the digits in the MNIST were checked manually, they still contain a residual error of 0.1% [Gro95]. The experiment was set up as follows: firstly, the digits were automatically pre-sorted using the CVL-Random set for training. Secondly, the participants manually corrected the results. However, this raises two problems. On the one hand, the digits may be broken due to the segmentation process. On the other hand, the ground truth is not intended to reflect the subjective perception of the reviewing person, but instead the intention of the writer. Therefore, an application was provided to the participants which highlights a given digit in the digit string it was extracted from. As the ground truth for the digit strings is provided by the instructions for the writers, this allows a correct labeling. Further, this way broken digits, as for instance parts of digits, can be identified. For these digits an additional 'broken' label is introduced. In Figure 4.7 an exemplary result of the application is shown for a digit '5'. All digits in the provided set, i.e. a superset of the CVL-HDRC test set which still contains different number of samples for each digit class, were labeled by both participants using the application.

To quantify the error for the manual labeling process, the labeling results of both participants are compared. This is achieved by computing the number of digits labeled equally and differently by both participants for each class. In Figure 4.8 and Table 4.3 the results are shown in absolute numbers and percentages for each class, respectively. The highest overlap is achieved for digits labeled as '1' with 98.42%. However, for the digit '9' the overlap is lowest with only 83.76%. The most problematic case are 'broken'

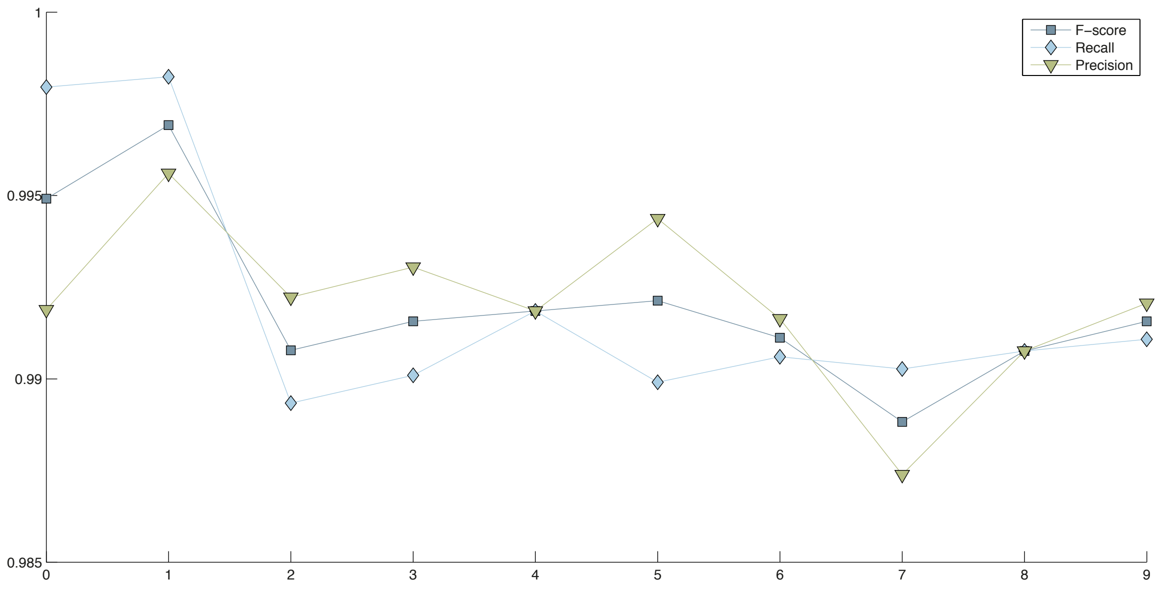


Figure 4.6: F-score, precision and recall for each digit class on the MNIST test set.

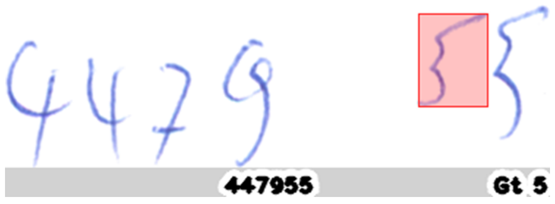


Figure 4.7: Result of the application highlighting a particular digit in the corresponding digit string. In the bottom row the ground truth for the digit string is shown (center) and the current label of the digit (right).

digits, as this is manly dependent on the individual choice of the participant. Therefore the participants only agreed on 23.17% of digits labeled as 'broken'.

4.1.5 Prediction Values

In order to allow a meaningful rejection of uncertain prediction results, prediction values are computed as described in Section 3.2.3. To evaluate the correlation of these values with the recognition accuracy an experiment is conducted as follows: in the first step, the digit recognition algorithm is trained using the CVL-HDRC training set. Secondly, the labels of the CVL-HDRC test set are computed and sorted by prediction value. The graph shown in Figure 4.9 is then subsequently formed by rejecting an increasing number of samples with minimal prediction values and computing the corresponding prediction accuracies.

The resulting function, rejection x $\mapsto$ accuracy f(x), has a steep slope at the beginning and slowly approaches an accuracy of 100% in the end. Although f is not monotonically increasing as rejecting a correctly classified sample leads to negative jumps, it can be fitted by a monotonically increasing exponential function $\hat{f}(x)$ by defining an $\epsilon = 0.001$ (0.1%)

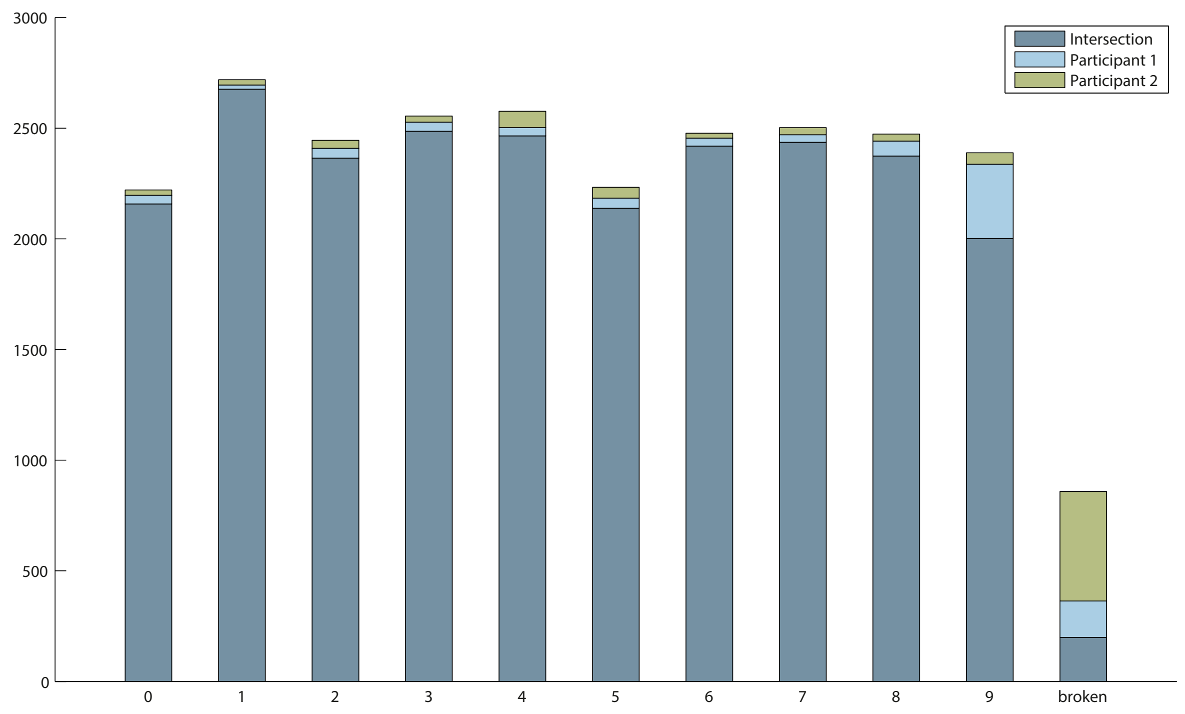


Figure 4.8: Absolute number of digits labeled as class C_i only by participant one (1), only by participant two (2) and by both participants (Intersection). The x-axis denotes the class label C_i and the y-axis the number of samples.

bound for the difference:

$$\hat{f}(x) = 1 - a_1 \cdot \exp\left(\frac{-a_2}{x}\right), \text{ with } \quad a_1 = 0.0630 \quad a_2 = 0.0974 \tag{4.5}$$

with the ϵ neighborhood defined by $|f - \hat{f}| < \epsilon$. The coefficients a_1 and a_2 were obtained in the fitting process. This shows, that although f is not monotonically increasing locally, the property holds globally. These results can be subsequently used to define thresholds t for the prediction values to guarantee a certain (statistical) accuracy. The thresholds for 95.0%, 99.0%, 99.5% and 99.9% are shown in Figure 4.9. However, as the function slows down as it approximates 100% the number of samples that have to be rejected to

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	$*$
$\cap$	97.16	98.42	96.73	97.30	95.69	95.75	97.66	97.36	96.00	83.76	23.17
$\boldsymbol{P_1}$	1.76	0.70	1.80	1.60	1.44	2.06	1.45	1.36	2.75	14.06	19.21
$\boldsymbol{P_2}$	1.08	0.88	1.47	1.10	2.87	2.19	0.89	1.28	1.25	2.18	57.63

Table 4.3: Relative number of digits labeled as class C_i only by participant one (P_1), only by participant two (P_2) and by both participants ($\cap$). The top row denotes the class labels C_i with the digits 0-9 and the 'broken' class ($*$). All values are relative (%) to the number of digits in each class.

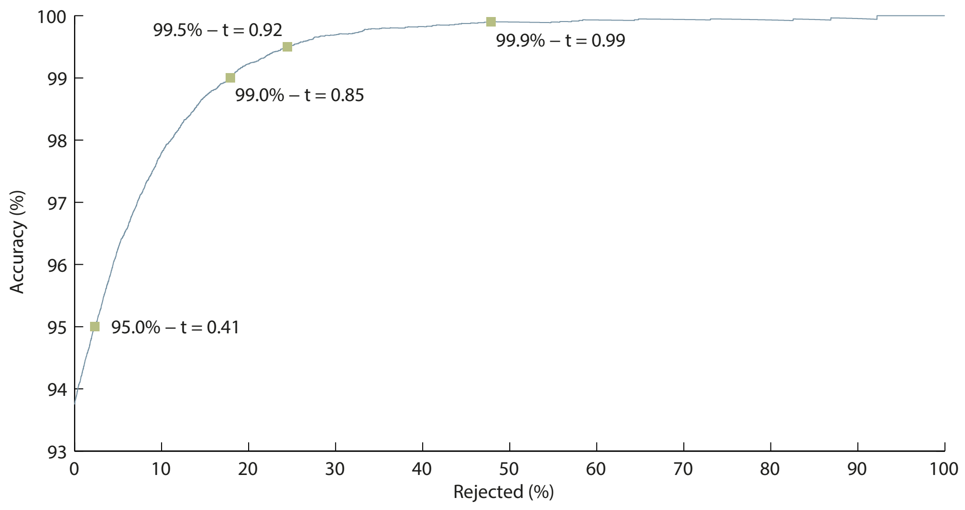


Figure 4.9: Digit recognition accuracy over the fraction of rejected samples for the CVL-HDRC test set. Further, the thresholds for the prediction values are highlighted for four accuracies.

reach a certain accuracy grows exponentially. For instance, to achieve 99.9% more than 40% of the samples have to be rejected. Even though, it shows that the prediction values provide means to identify digits with a higher probability of a false predicted class label.

4.2 Line Removal

In this section first the influence of spurious lines for the recognition performance is investigated. Secondly, the line removal algorithm proposed is evaluated using synthetically generated data from the CVL database.

4.2.1 Influence of Spurious Lines

As the digits are extracted from forms, handwritten strokes belonging to digits or sings may cross the form lines. To investigate the influence of spurious lines on the recognition accuracy, digits from the CVL database are synthetically degraded by placing horizontal and vertical lines in the images. The change in the recognition accuracy is then subsequently monitored for different placements and strengths of the synthetical line. This allows on the one hand an assessment of the influence of the lines and on the other hand, critical positions can be extracted to allow an evaluation of the line removal algorithm proposed in the worst case scenario (see Section 4.2.2).

Synthetical Line Placement

The lines with intensities $v \in [0, 1]$ and width w are placed at the relative position $p \in [0, 1]$ by multiplying with the graylevel image of a digit image $I(x, y)$. Assuming a line image

$L(x, y)$ with equal size as $I(x, y)$, it is defined as:

$$L(x, y) = \begin{cases} v & (x, y) \in \text{line} \\ 1 & \text{otherwise} \end{cases} \tag{4.6}$$

The spurious image $I_s(x, y)$ is computed by pixel-wise multiplication of the digit image and the line image:

$$I_s(x, y) = I(x, y) \cdot L(x, y) \tag{4.7}$$

In Figure 4.10 the results of this operation are shown for different line intensities and positions. From these results the assumption is made that adding vertical lines at the central position $p = 0.5$ has the most disruptive effect on the recognition accuracy.

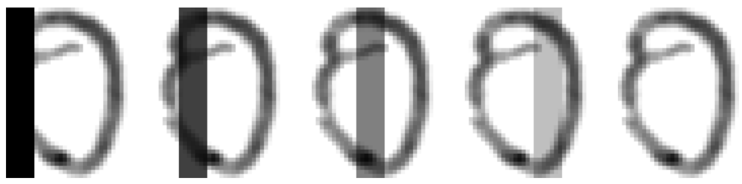


Figure 4.10: Synthetically added vertically lines with intensities and positions varying from 0 (left) to 1 (right) in 0.25 steps. Once the line intensity reaches 1 the line completely vanishes (right). The width is fixed to $w = 7$.

Evaluation

The precise impact on the recognition accuracy is subsequently evaluated using a grid of positions and intensities, i.e. $v = 0.1, 0.15, \ldots, 1$ and $p = 0, 0.05, \ldots, 1$. The width is fixed to $w = 7$ as this is the average width of the lines in the weather records. The experiment was conducted using the whole CVL-HDRC training set for training and 100 random samples of the validation set for evaluation. The results are shown in Figure 4.11.

As previously suspected, for vertical lines the placement at the central position achieves the lowest accuracy. For the minimum line intensity tested ($v = 0.1$) the recognition accuracy drops to 23% at this position, whereas the maximum accuracy for the same line intensity at $p = 0$ is 36%. However, for horizontal lines the lowest accuracy of 34% is achieved off-center at the positions $p = 0, 0.7$ and 0.75 with the same line intensity of $v = 0.1$. On the other hand, the highest accuracy for $v = 0.1$ is achieved at the central position $p = 0.5$ with 46%. This can be explained by the fact, that for instance in the digits '9' and '6' the information is more dense on the top and at the bottom, respectively. This is shown in Figure 4.12.

Overall this demonstrates, that spurious lines in the character images drastically decrease the recognition accuracy from over 90% to as low as 23%. Further, these results show that vertical lines placed in the digit images lead to larger drops in the accuracy than horizontal lines. An explanation for this is, that since on average the with of the digit is greater than its height, the vertical lines contain more pixels and therefore destroy more information in the digit images.

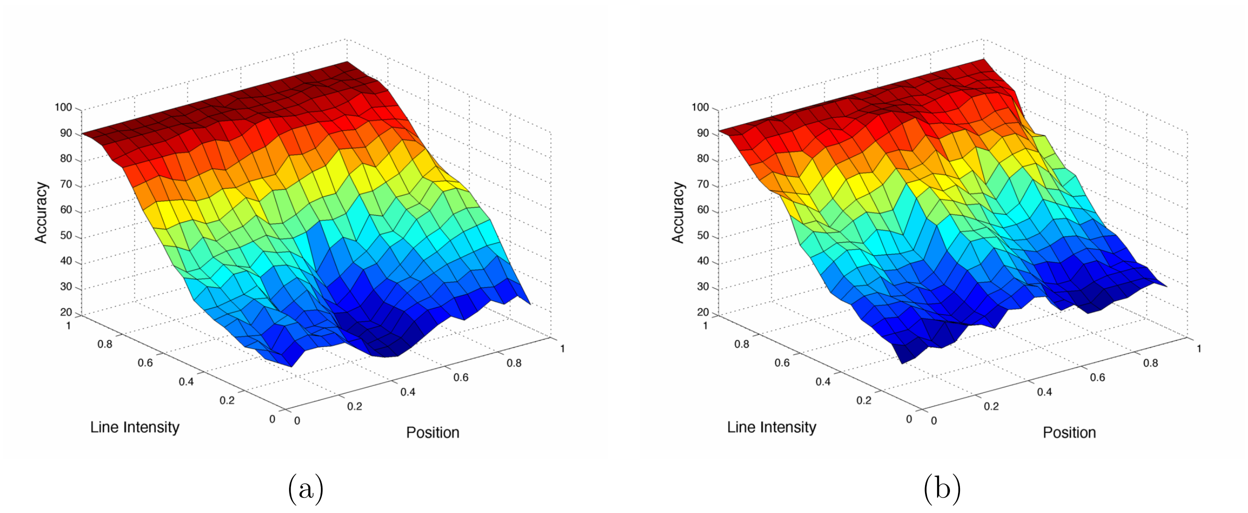


Figure 4.11: Surface plots of the digit recognition accuracy (in %) over the line intensity and position of synthetically placed lines. The intensities range from 1 (white) to 0 (black). The lines are placed either vertically (a) or horizontally (b) at positions ranging from 0 (left/top) to 1 (right/bottom), respectively.

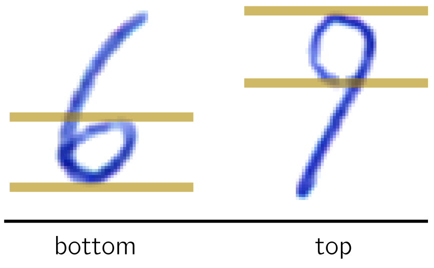


Figure 4.12: Areas where horizontal line placement is most destructive.

4.2.2 Stroke Preserving Line Removal

As shown in the previous section, spurious vertical or horizontal lines in the digit images lead to inferior classification results. To overcome this problem, a stroke preserving line removal approach was presented in Section 3.1.5. The goal of this section is to evaluate the strengths and weaknesses of this method.

In the previous section, the influence of the line intensity and position were examined and the worst case scenario for the line placement was identified. The placement at the center ($p = 0.5$) and at the bottom ($p = 0.7$) leads to the most inferior results for vertical and horizontal lines, respectively.

For this evaluation, the placement of the lines is carried out the same as in the previous section. However, only the positions $p = 0.5$ and $p = 0.7$ for the vertical and horizontal case, respectively, are considered. The width of the lines is again fixed to 7 pixels. Firstly, a naive approach of replacing the line pixels with background pixels is considered. Secondly, the stroke preserving line removal algorithm is applied on the whole digit image and thirdly, only pixels damaged by the line are replaced with the output of the algorithm.

This is similar to the approach used for the weather records and is reasonable, since as described in Section 3.1.5 the algorithm degrades the overall quality of the digit image. Additionally, the accuracy on digit images containing the synthetically placed lines is presented to quantify the performance gain.

For all approaches the normalization method described in Section 3.1.5 is applied. Additionally, for the stroke preserving line removal algorithm a padding of 15 pixels is added to each side of the image to ensure that the algorithm operates properly. The padding is afterwards removed by extracting the bounding box of foreground pixels. The experiment is conducted using 300 randomly selected digit images from the CVL-HDRC validation set and 21 line intensity values in the interval [0,1]. For training, the whole CVL-HDRC training set is used. In Figure 4.13 the results are depicted for vertical (left) and horizontal lines (right), respectively.

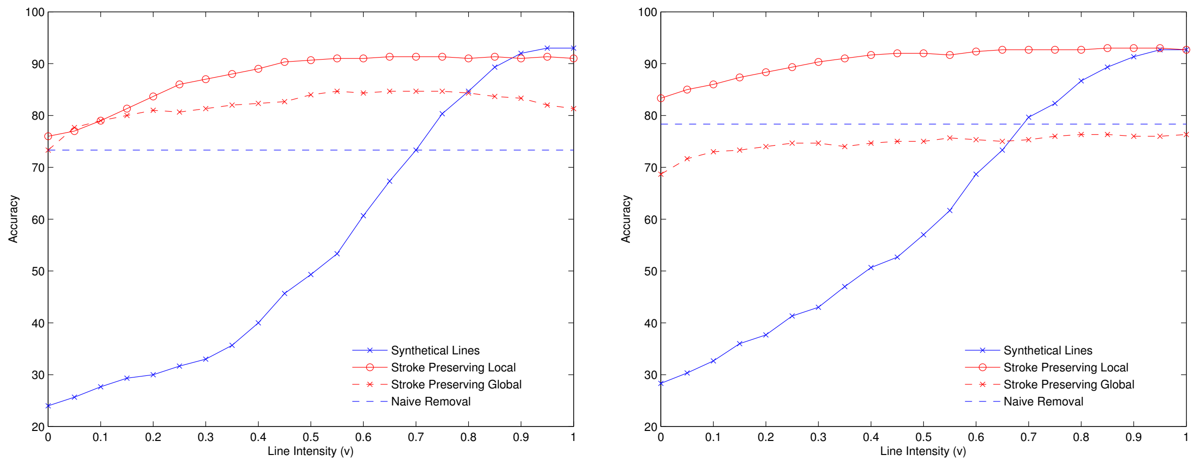


Figure 4.13: Evaluation of the digit recognition accuracy over line intensities. The position is fixed to $p = 0.5$ and $p = 0.7$ for the vertical case (left) and the horizontal case (right), respectively. The width of the lines is fixed to 7 pixels.

From these results several observations can be made: firstly, the naive approach is improving the accuracy for lines with low intensity values, i.e. black lines. However, for intensity values above about 0.7 the influence of the missing foreground pixels outweighs the impact of the spurious lines. Secondly, applying the stroke preserving line removal algorithm on the whole digit images to remove vertical lines leads to improved results compared to the naive approach. However, for horizontal lines it performs slightly worse. This suggests, that the vertical gradient is more influential for the digit recognition than the vertical since the digit images are reconstructed using only one gradient. In Figure 4.14 this is shown exemplarily on the digit '1'. Similarly to the naive approach, the accuracy values are significantly improved in comparison to the digit images containing spurious lines. Thirdly, replacing only areas of the image which are influenced by the lines with the result of the stroke preserving line removal method outperforms all other methods. For line intensities of over approximately 0.4-0.5 the maximal accuracy is achieved with only a slight or no performance drop compared to unaltered digit images. However, for dark lines there is still a performance difference of 10-15%. Further, these results show that a reliable detection of the lines is crucial to achieve the best accuracy possible.

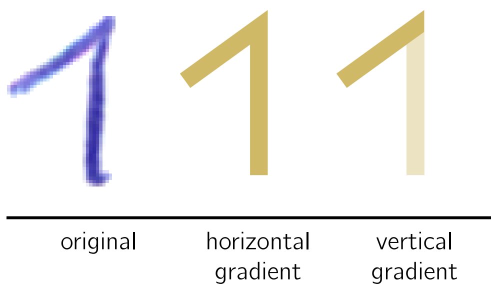


Figure 4.14: Schematics of the difference in information loss between the horizontal and vertical gradient of the digit '1'.

To investigate the influence of the width w, the same experiment is conducted using the width of the broadest lines in the weather records, i.e. $w = 11$. In Figure 4.15 the results are again depicted for vertical (left) and horizontal lines (right), respectively. Comparing this figure and Figure 4.14 it can be observed that the accuracy is overall decreased and in the worst case, i.e. digit images containing spurious vertical lines and line intensities lower than 0.4, it drops under 20%. Furthermore, due to the increased number of pixels influenced by the spurious lines, the performance of the naive approach is decreased to about 60% compared to about 70% for $w = 7$. Additionally, in contrast to line widths of $w = 7$, the stroke preserving line removal applied on the whole images outperforms the naive approach not only for vertical lines, but also for horizontal lines. For intensity values above about 0.7 the naive approach performs worse than the digit images still containing spurious lines which coincides with the experiments carried out with $w = 7$. Similarly, overall the stroke preserving line removal applied only on the damaged pixels again outperforms all other methods although the maximal accuracy is not achieved until intensity values above 0.7. Still, this shows that the proposed method is able to significantly improve the results even for wider lines.

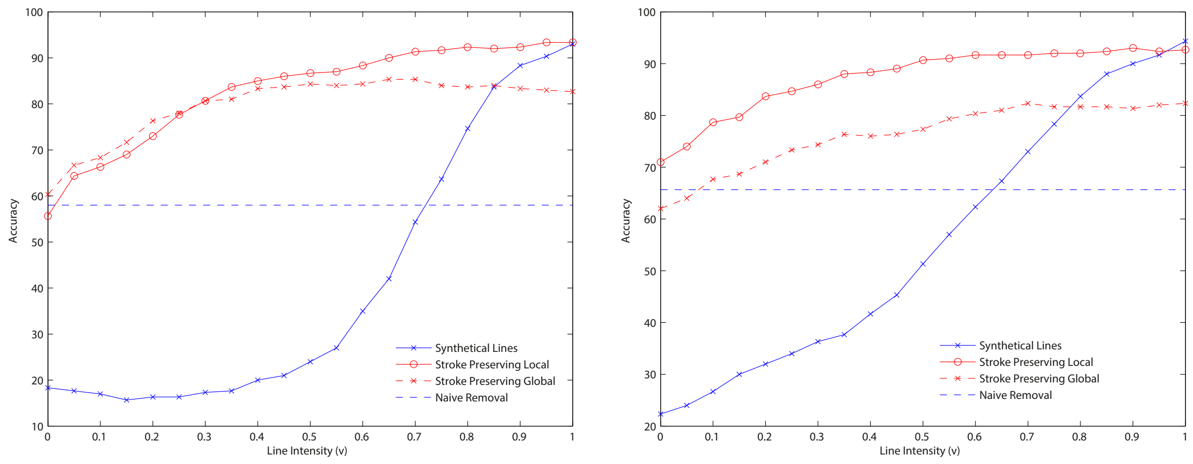


Figure 4.15: Evaluation of the digit recognition accuracy over line intensities. The position is fixed to $p = 0.5$ and $p = 0.7$ for the vertical case (left) and the horizontal case (right), respectively. The width of the lines is fixed to 11 pixels.

4.3 Weather Records

The evaluation of the whole process from scanned weather records to the classification of the digits and signs is conducted using a total number of 144 weather records from five different measurement stations. The complete set of records is divided into two different subsets:

Nasswald: 84 records from the measuring station “Nasswald” between 1994 and 2000. All records were created by the same writer. The months are sorted into a training set and a test set in an alternating manner, i.e. January, March, ... for the training set and February, April, ... for the test set.

5-Stations: 60 records from five measurement stations. From the Nasswald dataset only the year 1994 is included. The measurements from the other stations are from the year 2011. The training and test sets are created similarly as for the Nasswald dataset.

In this section, firstly the approach used for training the SVMs is described. Secondly, the evaluation results for the Nasswald dataset are presented. Thirdly, the 5-Stations dataset is evaluated and the results are compared.

4.3.1 Setup of the Experiments

The first step for training the digits and signs is the document analysis as described in Section 3.1. The only difference in the structure of the weather records regarded is the approximate vertical distance to the ROI which is set individually. The results of the document analysis are normalized images of the CCs assigned to cell positions in the table.

The ground truth used for training has disadvantages which have to be considered: firstly, it contains complete numbers (e.g. 0.23, 22.3 or -4.5). This means, that there is no information available about the leading and trailing digits and signs written down in the weather records. For instance, in some cases writers decided to omit the leading zero digit and in most of the cases the plus sign. This problem is further amplified due to the fact, that the segmentation of visually empty fields does not always lead to an empty segmented cell. One possibility to cope with this is to treat empty fields and zeros (or empty fields and plus signs) as one class. In case of the sign recognition this means, that all fields which are either empty or ‘+’ are trained as one class and all fields which should be ‘−’ are trained as another class. Of course it does not make sense to train fields which are empty after segmentation, therefore only fields which contain connected components are used for training. In case of the digit recognition, another possibility is exploited by adding an additional class for these empty fields, i.e. for cell images which are in any case empty in the ground truth. For example the temperature value ‘1.1’ can be assumed to contain an empty cell in front of the leading ‘1’. This strategy has proven to improve the results. For instance, using the years 1995 to 2000 of the Nasswald dataset for training and the year 1994 for testing, this improves the recognition accuracy from 98.26% to 99.10%.

However, in the prediction phase the 'empty' class and the class '0' are merged as they are equivalent for the assembly of the final number. The accuracy, confusion matrix, F-score, precision and recall are subsequently computed as described in Section 4.1.2.

4.3.2 Nasswald

There are several differences concerning the digit images between this dataset and the datasets evaluated in Section 4.1 which have to be considered. Firstly, the number of samples per class varies significantly. For instance, as shown in the last column in Table 4.4, the class '0' has 3,382 samples whereas the smallest class '9' only contains 521 samples. There are two reasons for this. On the one hand, the temperature values are within a limited range and therefore it is highly unlikely for digits greater than '3' to be present at the first position of a number. Similarly, values like 1*.* or 2*.* are very likely and therefore the number of samples in the classes '1' and '2' are higher. On the other hand, since empty fields and zeros are treated as one class the '0' class is obviously the largest. Secondly, due to the segmentation process artifacts which influence the training and prediction process may still be present in the digit images as shown in Figure 4.16.

		predicted class										
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	#
true class	**0**	**99.8**	·	0.1	·	·	·	·	·	·	·	3382
	1	0.1	**99.7**	0.1	·	·	·	·	·	·	·	2308
	2	0.2	·	**99.4**	0.2	·	·	·	·	0.1	0.2	1226
	3	0.2	·	0.1	**99.2**	0.1	0.1	·	·	·	0.2	914
	4	0.4	·	·	·	**99.6**	·	·	·	·	·	752
	5	0.2	·	·	0.3	0.2	**99.4**	·	·	·	·	658
	6	0.2	·	0.2	·	·	0.2	**99.5**	·	·	·	546
	7	·	·	·	0.2	0.2	·	·	**99.2**	·	0.5	610
	8	·	·	·	0.4	·	·	·	·	**99.5**	0.1	801
	9	0.2	·	·	0.4	0.6	0.2	·	·	0.4	**98.3**	521
	#	3388	2303	1226	917	756	658	543	606	801	520	**11718**

Table 4.4: Confusion matrix for the digit recognition on the Nasswald dataset. The rows indicate the actual class label extracted from the ground truth, whereas the columns indicate the prediction results. All values except the last row and column are given as fractions (in %) of the whole number of digit images per class. Values below 0.1% have been omitted for clarity. The diagonal represents the number of true positives. The last row and column represent the number of samples in each column and row, respectively.

The overall accuracy achieved on this dataset is 99.54% for the digit recognition and 99.52% for the sign recognition. As shown in Figure 4.17 the most problematic classes are '3' and '9' which have also proven to be problematic for the CVL-HDRC dataset. However the digit '8' performs significantly better in this dataset. Overall, the performance is significantly better than the for the CVL-HDRC dataset. The main reason for this is, that all digits are extracted from one writer which lowers the intraclass variability. Further, a

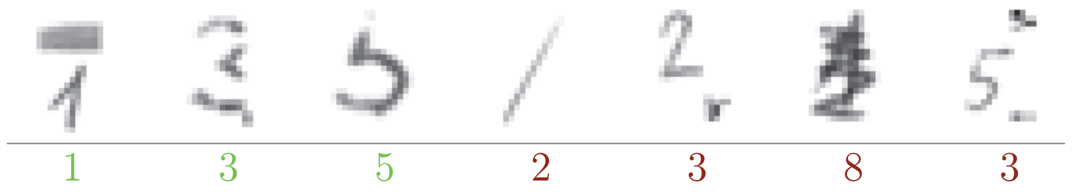


Figure 4.16: Examples of correctly (green) and falsely (red) classified normalized digit cells. The numbers at the bottom depicts the output of the prediction.

total number of 9,400 digits from one writer are available for training and additionally, empty fields are mostly discarded correctly in the segmentation process which increase the overall accuracy.

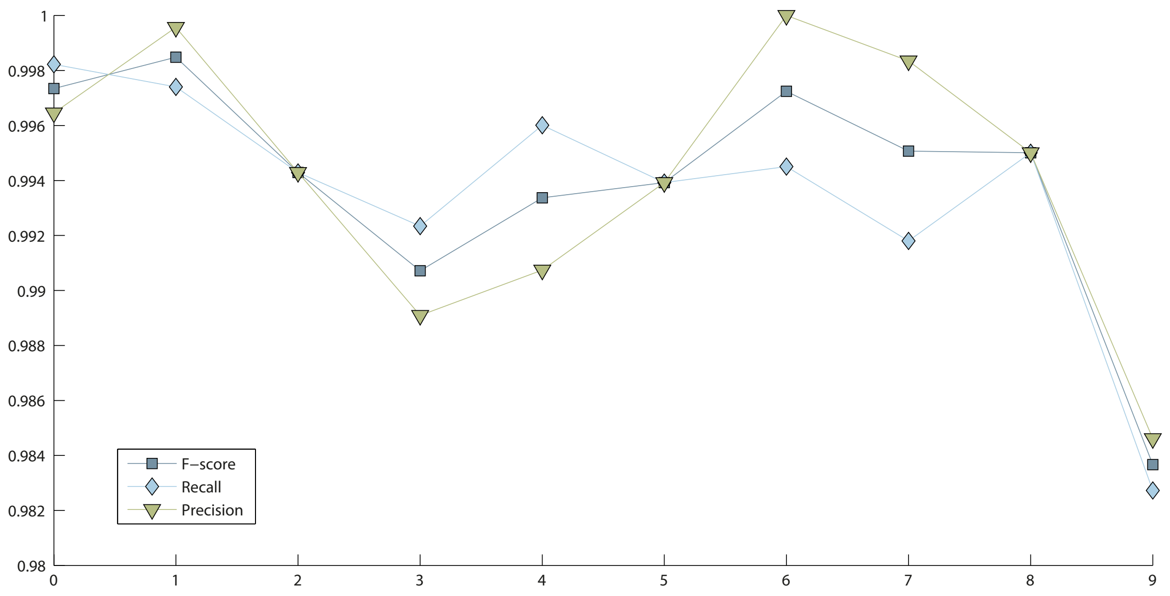


Figure 4.17: F-score, precision and recall for each digit class on the Nasswald test set.

The number of trained empty fields is very small in comparison - 65 or 0.69% compared to at least 500 for each digit class. From the 2,393 fields which were identified as empty using the ground truth, all except three were labeled the same in the classification. From the remaining 4 false positive empty fields two of them were falsely identified as empty by the document analysis whereas the other two were classified falsely due to underlinings assigned to the cell beneath as shown in Figure 4.16.

Overall, the main source of error for this dataset is the segmentation process. Firstly, artifacts not removed before the normalization stage lead to a degraded normalization of images. This is for instance shown in the 5th and 7th example in Figure 4.16 where the digits '2' and '5' are smaller and not centered due to artifacts. Secondly, parts of the digit may be missing if either the line removal algorithm is not able to preserve the handwritten strokes or in case of under-segmentation. However, the 2nd and 3rd digit in Figure 4.16 show that a correct classification is nevertheless possible. Other problems are underlinings of maxima and minima, corrections and meaningless strokes which are shown in the 1st, 4th and 6th example in Figure 4.16, respectively.

4.3.3 5-Stations

The overall digit recognition accuracy for the 5-Stations dataset is 87.9% and for sign recognition an accuracy of 95.2% is achieved. There are two main reasons for the performance drop compared to the Nasswald dataset. Firstly, this dataset contains at least five different writers as each measurement station is maintained by a different person. Therefore, on the one hand a smaller number of digits is trained per writers, and on the other hand the intraclass variability is increased due to different writing styles. Example images for this dataset are shown in Figure 4.18. It can be observed, that the strokes regularly cross cell borders and digits are even touching in some cases.

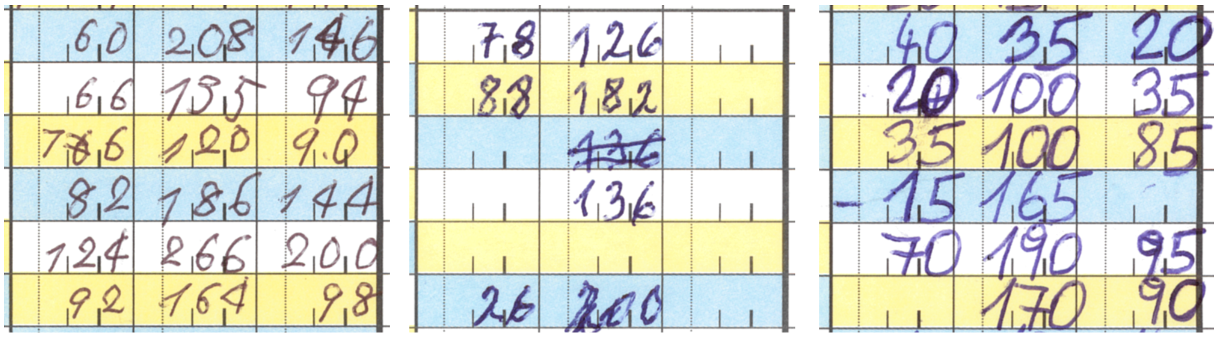


Figure 4.18: Details from the 5-Stations weather records with hard to recognize digits, digits crossing the cell borders and touching digits.

An indication for errors in the segmentation process is the confusion matrix in Table 4.5: the class ‘0’ achieves the second worst precision whereas for both the Nasswald and the CVL-HDRC dataset this class has shown high precision values with the best and second best precision of all classes, respectively. Since about 2/3 of this class reflect empty fields the drop in precision indicates errors in the segmentation process.

Further differences to the Nasswald dataset are visible when comparing Figure 4.17 and 4.19. However, it has to be kept in mind that the overall performance is significantly worse for the 5-Stations dataset. For instance the digits ‘3’ and ‘2’ perform better for the 5-Stations dataset. Overall, all datasets evaluated have in common that the worst performance is achieved for the class ‘9’.

To remove the small number of writers as a bias another experiment was conducted using the months from January to November for training and only December for testing. This way an accuracy of 93.98% for the digit recognition is achieved, which is significantly better than the results presented above. The remaining 5% difference to the Nasswald dataset can be accounted to the different writing styles and the segmentation process.

true class \ predicted class	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	#
0	**88.6**	3.1	1.5	1.1	1.0	1.1	1.2	0.8	0.9	0.7	3703
1	7.4	**91.8**	0.2	0.1	0.3	·	0.1	0.2	·	·	1091
2	7.6	0.9	**89.3**	0.6	0.1	0.2	0.3	0.3	0.6	·	860
3	6.1	·	0.7	**88.7**	0.2	2.8	·	0.2	0.7	0.5	426
4	8.3	·	0.5	0.2	**88.3**	1.0	0.5	0.7	·	0.5	420
5	5.3	0.2	·	1.1	0.4	**91.5**	0.4	0.2	0.4	0.4	470
6	8.3	0.2	1.0	0.2	0.2	0.7	**88.3**	·	0.5	0.5	409
7	6.4	·	·	0.3	2.0	1.0	·	**89.9**	·	0.3	298
8	6.7	·	1.0	1.0	0.2	0.7	0.2	0.5	**89.3**	0.5	419
9	10.6	0.4	0.7	0.4	1.5	1.8	0.7	2.6	1.5	**79.9**	274
#	3624	1126	839	436	428	504	416	317	425	255	**8370**

Table 4.5: Confusion matrix for the digit recognition on the 5-Stations dataset. The rows indicate the actual class label extracted from the ground truth, whereas the columns indicate the prediction results. All values except the last row and column are given as fractions (in %) of the whole number of digit images per class. Values below 0.1% have been omitted for clarity. The diagonal represents the number of true positives. The last row and column represent the number of samples in each column and row, respectively.

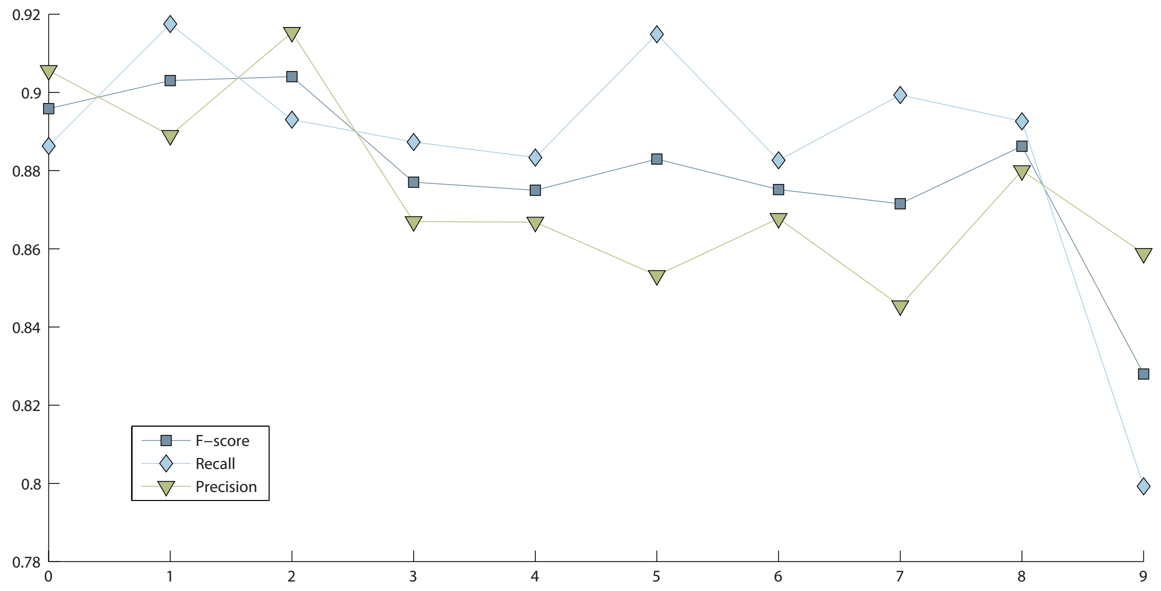


Figure 4.19: F-score, precision and recall for each digit class on the 5-Stations set.

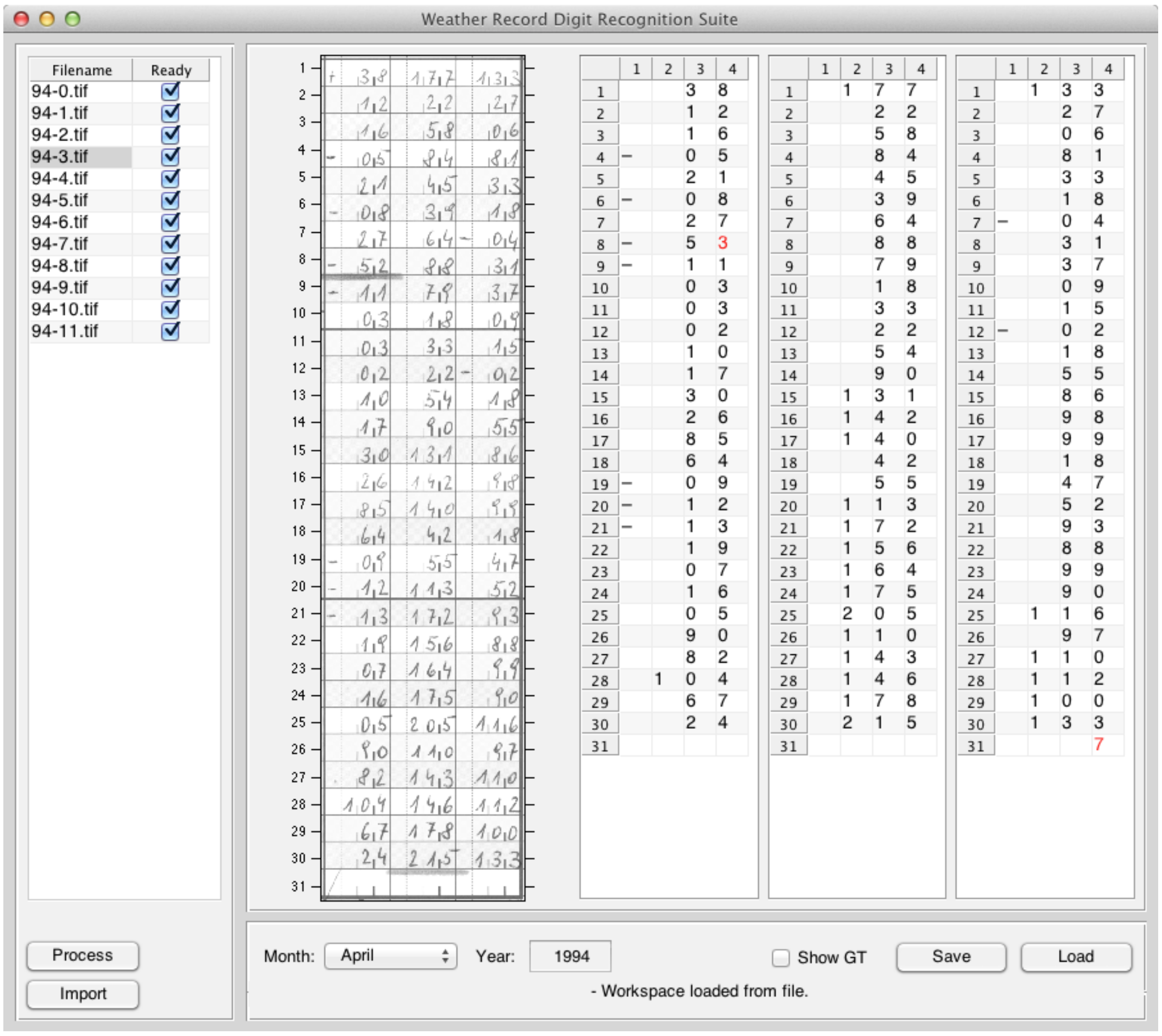
4.3.4 Prototype

In order visualize a possible work flow for the proposed automatic recognition of weather records a prototype was implemented. In the proposed application images of weather records can be imported, automatically processed and manually reviewed. The recognition of the digit and signs is carried out as described in Section 3.1 and the training of the SVMs is conducted as shown in Section 4.3.1. However, the application allows a manual reviewing of the computed results by displaying the ROI of the imported weather records. Furthermore, the prediction values are used to color-code the predicted class labels allowing an identification of uncertain results. Moreover, to visualize the accuracy achieved by the method proposed, the ground truth can be displayed - if available. On the one hand, this allows an intuitive estimation of the accuracy and on the other hand, challenging cases can be identified for further improvements. Additionally, the depicted results can be corrected and the whole workspace can be saved to a file. Since multiple weather records can be processed unsupervised, this allow for instance an automatic processing of a large number of records over night and a subsequent manual reviewing of the results in multiple sessions.

In Figure 4.20 the application is depicted with the results of an exemplary weather record. On the left side, the imported images can be selected for processing. Further, in the center the ROI of the current weather record is shown and on the right side the predicted digits and signs are presented with one table for each point of measurement, i.e. in the morning, at noon and in the evening. The first columns represent the signs and the remaining columns the two digits in front of and behind the decimal point, respectively. In the bottom area the date of the record can be set to allow the visualization of the ground truth, i.e. correct values are displayed in black and false values in red. Low prediction values are similarly visualized using the colors red or yellow, i.e. if predicted digits or signs are identified as uncertain. For instance the digit '3' from the measurement on the 8th in the morning is uncertain as it is influenced by an underlining. Furthermore, the digit '7' on the 31st in the evening is similarly highlighted as uncertain. By comparing the recognition results to the image of the ROI in the center it can be observed that both highlighted digits are wrong, i.e. '3' should be '2' and '7' should be 'empty'. The results can subsequently be manually corrected by changing these numbers in the respective tables.

Summary

In this chapter the system presented was evaluated using three different digit databases, synthetically generated data and weather records from five different measurement stations. It was shown that an recognition accuracy of over 99% can be achieved on the MNIST database. Further, the CVL digit database was used to investigate the challenges provided by different digit classes. Therefore, F-score, precision and recall were described and computed. It was shown that the digits '8' and '9' are the most challenging. Moreover, the human error was investigated in an experiment involving the manual annotation of digit images. Additionally, the correlation of the prediction values to the prediction accuracy was depicted.



Weather Record Digit Recognition Suite

Filename	Ready
94-0.tif	✓
94-1.tif	✓
94-2.tif	✓
94-3.tif	✓
94-4.tif	✓
94-5.tif	✓
94-6.tif	✓
94-7.tif	✓
94-8.tif	✓
94-9.tif	✓
94-10.tif	✓
94-11.tif	✓

	1	2	3	4
1			3	8
2			1	2
3			1	6
4	–		0	5
5			2	1
6	–		0	8
7			2	7
8	–		5	3
9	–		1	1
10			0	3
11			0	3
12			0	2
13			1	0
14			1	7
15			3	0
16			2	6
17			8	5
18			6	4
19	–		0	9
20	–		1	2
21	–		1	3
22			1	9
23			0	7
24			1	6
25			0	5
26			9	0
27			8	2
28		1	0	4
29			6	7
30			2	4
31				

	1	2	3	4
1		1	7	7
2			2	2
3			5	8
4			8	4
5			4	5
6			3	9
7			6	4
8			8	8
9			7	9
10			1	8
11			3	3
12			2	2
13			5	4
14			9	0
15		1	3	1
16		1	4	2
17		1	4	0
18			4	2
19			5	5
20		1	1	3
21		1	7	2
22		1	5	6
23		1	6	4
24		1	7	5
25		2	0	5
26		1	1	0
27		1	4	3
28		1	4	6
29		1	7	8
30		2	1	5
31				

	1	2	3	4
1		1	3	3
2			2	7
3			0	6
4			8	1
5			3	3
6			1	8
7	–		0	4
8			3	1
9			3	7
10			0	9
11			1	5
12	–		0	2
13			1	8
14			5	5
15			8	6
16			9	8
17			9	9
18			1	8
19			4	7
20			5	2
21			9	3
22			8	8
23			9	9
24			9	0
25		1	1	6
26			9	7
27		1	1	0
28		1	1	2
29		1	0	0
30		1	3	3
31				7

Month: April Year: 1994 Show GT

- Workspace loaded from file.

Figure 4.20: Prototype for the visualization of a semi-automatic processing of weather records. On the left, images of weather records can be loaded and automatically processed. In the center, the ROI of the current record is depicted and on the right the recognition results are shown. The controls in the bottom area are used to visualize the ground truth and save/load the whole workspace. The weather records depicted are from the measurement station Nasswald from the year 1994.

For the investigation of the influence of spurious horizontal and vertical lines, digit images from the CVL database were synthetically degraded by placing lines with varying intensities at multiple positions. It was shown that spurious lines can decrease the recognition accuracy by 75% in the worst case. Further, an experiment was conducted which demonstrated the performance improvement of using the proposed stroke preserving line removal approach.

Finally, the entire system was tested using two sets of weather records. The results show, that an accuracy of 93% can be achieved on weather records from at least five different writers. Furthermore, by restricting the dataset to records from a single writer the results could be improved to over 99%. Although, these results are promising, still an residual error of 1-2 wrongly predicted digit per month remains. Therefore, an application was presented which allows the manual review and correction of the predicted digits and signs. Since the prediction values are used to emphasize uncertain results, the reviewing effort is greatly reduced compared to the current approach of manually transcribing the weather records without automatic assistance.

Chapter 5

Conclusion

In this report an approach for the automatic recognition of temperature values in weather records is introduced. The segmentation of the digits is achieved by reconstructing the tabular structure using line detection. Further, to reduce the influence of spurious lines a stroke preserving line removal method is proposed. In contrast to other approaches which apply heuristics on binary images, it is based on the reconstruction of gradient images by applying a Wiener filter. Using a trained PCA basis as a filterbank the features of the digits and signs are extracted and subsequently classified with multiple SVMs having RBF kernels. Moreover, to allow an identification of uncertain prediction results, prediction values are computed to estimate the class probabilities. The evaluation was conducted using three different digit databases with manually annotated ground truth, synthetically generated digit images with spurious lines and weather records from five different measurement stations located in Lower Austria with manually transcribed temperature values.

Experiments on handwritten digit databases were conducted to depict the digit recognition performance for the best case scenario of manually segmented images. Further, the influence of disjoint writers in the training and test set were investigated using the CVL database as it allows an identification of the digits' writers. Additionally, it was shown which digit classes are frequently confused and the recognition performance for the individual classes was investigated. Furthermore, the validity of the prediction values was depicted by investigating their correlation to the prediction accuracy. To quantify the human error of a manual annotation of handwritten digits, the annotation results of two participants were compared.

To identify the influence of spurious lines on the digit recognition accuracy, synthetically generated horizontal and vertical lines with varying intensity were placed at different positions in the digit images from the CVL database. It was shown that these line can lead to a decrease in recognition accuracy of up to 75%. Simultaneously the worst case scenarios for the line positions were identified and the performance gain of using the stroke preserving line removal method proposed was shown for these cases. At the same time the limitations of this approach were investigated.

For the evaluation of the whole system weather records from five different measurement stations were used to create two different datasets. One dataset, containing records from the measuring station Nasswald, was used to depict the performance of the system for data from a single writer. The second dataset on the other hand was used to identify the performance difference to data from multiple writers. Furthermore, the results for the individual digit classes were compared to their performance achieved on the handwritten digit databases. Additionally, the influence of artifacts resulting from the segmentation process were investigated. To depict an exemplary work flow, an application was implemented which allows the manual review and correction of the predicted digits and signs.

Disadvantages of the Proposed System

The evaluation on the handwritten digit databases in Section 4.1 showed that although an accuracy of over 99% is achieved on the MNIST database the accuracy drops to about 93% for the CVL database which is closer to real world scenarios. Furthermore, although the proposed method for stroke preserving line removal is able to retain the recognition accuracy for digit images with spurious lines in most cases, lines with low intensities (black) are still challenging and lead to a drop in accuracy of 10-15% as shown in Section 4.2.2. In these cases there is no difference in the intensity values of the lines and strokes and therefore no gradient information which can be exploited. Moreover the results show that a localization of the lines is crucial.

Furthermore, the experiments conducted using the weather records in Section 4.3 show that compared to a single writer multiple writers are still challenging. However, the most frequent errors are introduced by the segmentation process. Due to artifacts remaining after the segmentation the digits are not properly normalized which decreases the recognition accuracy. Furthermore, digits which are not written in their respective cells raise problems as the localization of the digits is based on the reconstructed tabular structure of the forms. Additionally, the segmentation produces incorrect results when challenged with touching characters as the segmentation of digit strings is not treated in the system proposed. Moreover, meaningless strokes and corrections can not be handled by the system. Besides that, underlinings may produce incorrect results. As they are not treated separately and as there is only one maxima and minima per month, there are too little samples for the digit recognition to adapt.

Advantages of the Proposed System

As stated in Section 4.1 the proposed digit recognition achieves an accuracy of 93% to 99% on handwritten digit databases. Further, when digits of the writers in the test set are included in the training, the accuracy is for instance on the CVL database improved from about 93% to 96%. Moreover, when compared to a manual annotation of handwritten digits these results are promising, as the overlap of the two peoples participating in the experiment shown in Section 4.1.4 ranges between 83% and 98%. However, it has to be noted that this only provides an estimation of the human error as it may vary significantly depending on the individual commitment to the task. Additionally, it was shown in Section 4.1.5 that the computed prediction values provide means to identify uncertain prediction results, i.e. results with a high probability of being wrong. This allows for instance the design of a semi-automatic system in which uncertain results are highlighted for manual correction.

Furthermore, as stated in Section 4.2.2 the proposed stroke preserving line removal greatly improves the recognition accuracy compared to digit images still containing spurious lines. Moreover, for lines with intensities of above 0.4-0.5 the performance is similar to the recognition of undegraded images.

Additionally, the experiments conducted in Section 4.3 show that the entire system achieves an accuracy of over 99% per digit on a dataset with weather records from a single measurement station. Although, the dataset containing records of five different stations is more challenging due to touching characters, corrections, multiple writers, etc., still an accuracy of 93% per digit is achieved when trained with a sufficient number of samples. This is comparable to the results on the CVL digit database. Furthermore, the proposed prototype shows that the remaining errors can efficiently be corrected by a manual reviewing process.

Future Work

To cope with intraclass variability in the digit images, on the one hand, slant normalization could be used to reduce it [LNSF04]. On the other hand, using affine transformations and elastic distortions the training set could be increase in order to create a digit recognition system which is insensitive to these deformation [CMGS10].

For the entire system, the main performance enhancements are expected by making the segmentation process more robust to artifacts. Further, employing a digit string segmentation would on the one hand solve the problems arising from touching digits and additionally, it would allow the automatic transcription of weather records without separated digit cells. Moreover the stroke preserving line removal algorithm could be extended to allow the removal of underlinings. To treat lines with low intensities where no gradient information is available, the color information could be exploited since in most cases the stroke's color differs from the printed line. Moreover, the performance of the recognition of weather records from multiple writers could be improved by a prior identification of the writer.

Nomenclature

CC	Connected Component
SVM	Support Vector Machine
NN	Neural Networks
K-NN	k-Nearest Neighbor
OCR	Optical Character Recognition
CNN	Convolutional Neural Network
RVM	Relevance Vector Machine
MLP	Multi-Layer Perceptron
RBF	Radial Basis Function
PCA	Principle Component Analysis
MQDF	Modified Quadratic Discriminant Function
IDM	Image Distortion Model
DSCC	Directional Single-Connected Chain
ROI	Region of Interest
NIST	National Institute of Standards and Technology

Bibliography

[AKR07] K.R. Arvind, J. Kumar, and A.G. Ramakrishnan. Line removal and restoration of handwritten strokes. In *Proceedings of the International Conference on Computational Intelligence and Multimedia Applications*, volume 3, pages 208 –214, Dec 2007.

[AYV01] N. Arica and F.T. Yarman-Vural. An overview of character recognition focused on off-line handwriting. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 31(2):216 –233, May 2001.

[BBZS09] M. Bulacu, A. Brink, T. Zant, and L. Schomaker. Recognition of handwritten numerical fields in a large single-writer historical collection. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 808 –812, Jul 2009.

[BDL$^+$10] Rudolf Brázdil, Petr Dobrovolný, Jürg Luterbacher, Anders Moberg, Christian Pfister, Dennis Wheeler, and Eduardo Zorita. European climate of the past 500 years: New challenges for historical climatology. *Climatic Change*, 101:7–40, 2010.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[BJH$^+$10] Reinhard Böhm, Philip Jones, Johann Hiebl, David Frank, Michele Brunetti, and Maurizio Maugeri. The early instrumental warm-bias: A solution for long central european temperature series 1760–2007. *Climatic Change*, 101:41–67, 2010.

[BKH$^+$06] P. Brohan, J. J. Kennedy, I. Harris, S. F. B. Tett, and P. D. Jones. Uncertainty estimates in regional and global observed temperature changes: A new data set from 1850. *J. Geophys. Res.*, 111(D12), Jun 2006.

[Bre94] Thomas M. Breuel. A system for the off-line recognition of handwritten text. Technical Report Idiap-RR-02-1994, IDIAP, 1994.

[Bur98] J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[CCMM98] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: A review. Technical Report ITC-irst TR9703-09, ITC-irst, 1998.

[CKLS07] Mohammed Cheriet, Nawwaf Kharma, Cheng-lin Liu, and Ching Suen. *Character Recognition Systems: A Guide for Students and Practitioners.* Wiley-Interscience, 2007.

[CL11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, May 2011.

[CMGS10] Dan Claudiu Cireşan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12):3207–3220, Dec 2010.

[CMS12] Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. Technical Report IDSIA-04-12, IDSIA, 2012.

[DKS12] M. Diem, F. Kleber, and R. Sablatnig. Skew estimation of sparsely inscribed document fragments. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 292–296, 2012.

[Gar96] Michael D. Garris. Method and evaluation of character stroke preservation on handprint recognition. In *Proceedings of Document Recognition III*, volume 2660, pages 321–332, National Institute of Standards and Technology, 1996. SPIE.

[GBC$^+$95] M.D. Garris, J.L. Blue, G.T. Candela, D.L. Dimmick, J. Geist, P.J. Grother, S.A. Janet, and C.L. Wilson. Off-line handwriting recognition from forms. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2783 –2788, Oct 1995.

[GBC$^+$97] Michael D. Garris, James L. Blue, Gerald T. Candela, Gerald T. C, Patrick J. Grother, Stanley A. Janet, and Charles L. Wilson. NIST form-based handprint recognition system (release 2.0). Technical Report NISTIR 5959, NIST, 1997.

[GLF$^+$09] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855 –868, May 2009.

[Gro95] P. J. Grother. NIST special database 19: Handprinted forms and character database. Technical report, NIST, 1995.

[GW08] R. C. Gonzalez and R. E. Woods. *Digital Image Processing.* Person Prentice Hall, 3rd edition, 2008.

[Hei88] Richard Heim. Personal computers, weather observations, and the national climatic data center. *Bulletin of the American Meteorological Society*, 69(5):490–495, May 1988.

[Hul94] J.J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

[KDGN07] D. Keysers, T. Deselaers, C. Gollan, and H. Ney. Deformation models for image recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1422 –1435, Aug 2007.

[KL03] S. Sathiya Keerthi and Chih-Jen Lin. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation*, 15(7):1667–1689, Jul 2003.

[LBBH98] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278 – 2324, Nov 1998.

[LBM08] K. Labusch, E. Barth, and T. Martinetz. Simple method for high-performance digit recognition based on sparse coding. *IEEE Transactions on Neural Networks*, 19(11):1985 –1989, Nov 2008.

[LC] Y. LeCun and C. Cortes. The mnist database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

[LD05] Hailong Liu and Xiaoqing Ding. Handwritten character recognition using gradient feature and quadratic classifier with multiple discrimination schemes. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 19 – 23, 2005.

[LDW95] Jinhui Liu, Xiaoqing Ding, and Youshou Wu. Description and recognition of form and automated form data entry. In *Proceedings of the International Conference on Document Analysis and Recognition*, volume 2, pages 579 –582, Aug 1995.

[LNSF03] Cheng-Lin Liu, Kazuki Nakashima, Hiroshi Sako, and Hiromichi Fujisawa. Handwritten digit recognition: Benchmarking of state-of-the-art techniques. *Pattern Recognition*, 36(10):2271 – 2285, 2003.

[LNSF04] Cheng-Lin Liu, Kazuki Nakashima, Hiroshi Sako, and Hiromichi Fujisawa. Handwritten digit recognition: Investigation of normalization and feature extraction techniques. *Pattern Recognition*, 37(2):265 – 279, 2004.

[Man74] Gordon Manley. Central england temperatures: Monthly means 1659 to 1973. *Quarterly Journal of the Royal Meteorological Society*, 100(425):389–405, 1974.

[MNY99] Shunji Mori, Hirobumi Nishida, and Hiromitsu Yamada. *Optical Character Recognition.* John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1999.

[NB06] H. Nielson and W. Barrett. Consensus-based table form recognition of low-quality historical documents. *International Journal of Document Analysis and Recognition*, 8:183–200, 2006.

[Ots79] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[Pfi01] C Pfister. Klimawandel in der Geschichte Europas. Zur Entwicklung und zum Potenial der Historischen Klimatologie. *Österreichische Zeitschrift für Geschichtswissenschaften*, 12(2):7–43, 2001.

[PG08] Rafael Palacios and Amar Gupta. A system for processing handwritten bank checks automatically. *Image and Vision Computing*, 26(10):1297 – 1313, 2008.

[PS00] R. Plamondon and S.N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):63 –84, Jan 2000.

[RVF12] J. Richarz, S. Vajda, and G.A. Fink. Towards semi-supervised transcription of handwritten historical weather reports. In *Proceedings of the International Workshop on Document Analysis Systems*, pages 180 –184, March 2012.

[Sav98] A.E. Savakis. Adaptive document image thresholding using foreground and background clustering. In *Proceedings of the International Conference on Image Processing*, volume 3, pages 785 –789, Oct 1998.

[SGS93] S.N. Srihari, V. Govindaraju, and A. Shekhawat. Interpretation of handwritten addresses in US mailstream. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 291 –294, Oct 1993.

[SLT10] B. Su, S. Lu, and C.L. Tan. Document image binarization using background estimation and stroke edges. *International Journal on Document Analysis and Recognition*, 13:303–314, 2010.

[SLT11] B. Su, S. Lu, and C.L. Tan. Combination of document image binarization techniques. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 22–26, 2011.

[Sri06] S. Srihari. Handwriting recognition, automatic. In *Encyclopedia of Language & Linguistics*, pages 203 – 211. Elsevier, Oxford, second edition, 2006.

[TJT96] Øivind Due Trier, Anil K. Jain, and Torfinn Taxt. Feature extraction methods for character recognition - A survey. *Pattern Recognition*, 29(4):641 – 662, 1996.

[TL02] Loo-Nin Teow and Kia-Fock Loe. Robust vision-based features and classification schemes for off-line handwritten digit recognition. *Pattern Recognition*, 35(11):2355 – 2364, 2002.

[Vap99] V.N. Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5):988 –999, Sep 1999.

[ZC89] Jiacheng Zhang and Thomas J. Crowley. Historical climate records in china and reconstruction of past climates. *Journal of Climate*, 2(8):833–849, May 1989.

[ZLDP01] Yefeng Zheng, Changsong Liu, Xiaoqing Ding, and Shiyan Pan. Form frame line detection with directional single-connected chain. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 699 –703, 2001.