Maschinelle Erkennung handgeschriebener Zahlen

Diplomarbeit

der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Matthias Zimmermann

1996

Leiter der Arbeit:
Prof. Dr. Horst Bunke
Institut für Informatik und angewandte Mathematik

Zusammenfassung

Die maschinelle Handschrifterkennung ist bereits seit längerer Zeit ein Gebiet intensiver Forschung. Mögliche Anwendungsbereiche bilden die Erkennung und Verarbeitung von Dokumenten wie Postadressen oder Zahlungsbelege. Ein Teilgebiet der maschinellen Handschrifterkennung stellt die Erkennung handgeschriebener Zahlen dar. Falls nur das Schriftbild beurteilt wird und Information zum Schreibvorgang fehlen, spricht man von off-line Erkennung.

Die vorliegende Arbeit stellt ein System vor, welches das off-line Erkennen handgeschriebener Zahlen ermöglicht. Das System kombiniert einen Ziffernerkenner mit einer strukturellen Methode zur Segmentierung von Zahlen und erlaubt die Rückweisung von problematischen Eingaben. In Experimenten mit Zahlen der NIST SD3 Datenbank erkannte das System 73% der Zahlen bei einer Fehlerrate von 0.5%.

Verdankungen

Für die Leitung dieser Arbeit bedanke ich mich bei Herrn Prof. Horst Bunke. Speziell danke ich Dr. Thien Ha Minh. Durch ihn lernte ich eine systematische und fachlich fundierte Vorgehensweise in wissenschaftlichen Projekten zu schätzen.

Bei Gérard Rumo C34 und dem Zahlungsverkehr der POST bedanke ich mich für die Zusammenarbeit und das wertvolle Bildmaterial, das uns zur Verfügung gestellt worden ist.

Für die ungezählten, gemütlichen Kaffeepausen danke ich Urs-Viktor Marti, und für das aufwendige Korrekturlesen dieser Dokumentation geht mein Dank an Guido Kaufmann und Marcel Zumbühl. Diese Personen haben ausserdem viel zu meinem sozialen Umfeld beigetragen.

Ebenso danke ich an dieser Stelle meinen Eltern, die mir durch ihre Unterstützung dieses Studium ermöglicht haben.

Zum Schluss danke ich meiner Freundin Regula Weber für ihren grossen Beitrag an mein Lebensglück.

Bern, Oktober 1996 Matthias Zimmermann

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Dass Abbildung 1.1 die Zahl '32780' darstellt, ist für die Leserin beziehungsweise den Leser dieser Arbeit wahrscheinlich auf den ersten Blick offensichtlich. Unternimmt man jedoch den Versuch, genau zu erklären, was sich beim Lesen im eigenen Kopf abgespielt hat, beginnt man zu verstehen, dass solche Abläufe ausserordentlich komplex sind.

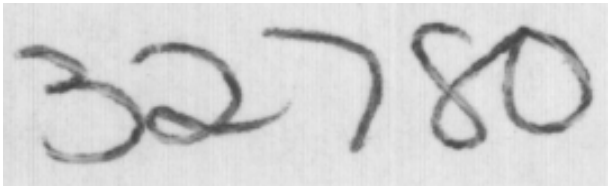


Abbildung 1.1: Das Bild einer Zahl

Um obiges Bild zu erkennen, müssen die verschiedensten Aufgabenstellungen gelöst werden: Die Differenzierung von Vorder- und Hintergrund, die Aufteilung des Vordergrundes in Teilstücke, die Gruppierung oder weitere Aufteilung von Teilstücken, die Erkennung solcher Regionen anhand gespeicherten Wissens, usw.

Da uns Erkennungsvorgänge im menschlichen Gehirn nur sehr beschränkt zugänglich sind, kennen wir auch die leistungsfähigen "biologischen" Verfahren nicht, die dahinter stecken. Es bleibt uns also nichts anderes übrig, als die unterschiedlichsten Ideen zu prüfen, um die Problematik der Erkennung von Handschrift besser zu verstehen.

1.1 Ausgangslage und Ziel

Bereits seit einigen Jahren beschäftigt man sich in der Fachgruppe für künstliche Intelligenz am IAM[1] der Universität Bern mit der Erkennung von Handschrift. Neben Gebieten wie der Erkennung von Ziffern, Buchstaben, Zahlen und Wörtern wurden auch verwandte Fragestellungen wie die Kombination von Erkennern untersucht.

Im November 1994 hat Dieter Niggeler unter der Leitung von Prof. Horst Bunke und Dr. Thien Ha Minh seine Diplomarbeit [Nig94] am IAM auf dem Gebiet der Handschrifterkennung beendet. Während dieser Arbeit ist ein System zur Erkennung handgeschriebener Zahlen entstanden.

Das Ziel der vorliegenden Diplomarbeit bestand darin, die Arbeit von Dieter Niggeler weiterzuführen und abzuklären, wie sich das vorgeschlagene Erkennungssystem verbessern, beziehungsweise ergänzen lässt. Ausserdem soll das System mit aktuellen, publizierten Forschungsarbeiten bezüglich Erkennungsresultate verglichen werden.

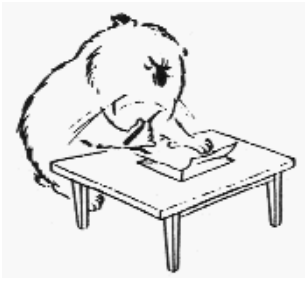
1.2 Gliederung der Dokumentation

Das Kapitel 2 behandelt die allgemeine Problematik der Handschriftterkennung sowie deren Teilgebiete. In Kapitel 3 werden die verschiedenen Datenbanken vorgestellt, die das verwendete Bildmaterial beinhalten.

Um die generellen Konzepte eines Erkennungssystems zu erläutern, wird im Kapitel 4 ein Ziffernerkenner vorgestellt. Das System zur Erkennung von Ziffernpaaren, das während dieser Diplomarbeit entstanden ist, wird in Kapitel 5 beschrieben.

Kapitel 6 dokumentiert das gesamte Erkennungssystem für handgeschriebene Zahlen. Experimente, die mit diesem System durchgeführt wurden, sind mit den zugehörigen Resultaten im Kapitel 7 dokumentiert. In Kapitel 8 werden die erreichten Resultate diskutiert und mit publizierten Resultaten von anderen Methoden verglichen. Schlussfolgerungen werden in Kapitel 9 gezogen.

[1] Institut für Informatik und angewandte Mathematik



Kapitel 2

Handschrifterkennung

Dieses Kapitel geht im Abschnitt 2.1 zuerst auf die allgemeine Problematik der Handschrifterkennung ein. Ihre Einordnung in der Informatik und Aufteilung in Teilgebiete wird in Abschnitt 2.2 behandelt. Da die maschinelle Erkennung von Handschrift nicht nur von akademischem Interesse ist, führt Abschnitt 2.3 zwei mögliche Anwendungen auf.

2.1 Problematik

Durch die maschinelle Erkennung von Handschrift wird eine sehr einfache Form der Kommunikation zwischen Menschen und Maschinen möglich. Papier und Schreibzeug ist alles was für diese Art der Kommunikation auf der Seite des Menschen notwendig ist.

Auf der Seite des Computers müssen Erfassungsgeräte wie Scanner, Ausgabegeräte wie Drucker oder Monitore sowie die notwendige Software vorhanden sein. Bei der Erfassung eines handschriftlichen Dokumentes fallen jedoch Unmengen von Daten an. Zur Erfassung von Abbildung 1.1 mussten die Helligkeitswerte von über 200000 Bildpunkte gemessen werden. Diese werden im Computer gespeichert und anschliessend verarbeitet. Bei der Erkennung von solchen Bildern geht es also darum, riesigen Datenmengen ihre Bedeutung zu entlocken.

Die enorme Vielfalt der Handschriften stellt die Hauptproblematik maschineller Handschrifterkennung dar. Folgende Faktoren können auf die Handschrift einen Einfluss haben: Die Ausbildung und das soziale Umfeld der schreibenden Person, die Körperhaltung, das benutzte Schreibwerkzeug, die Schreibunterlage usw. Dieser Problematik kann mit einer geschickten Wahl beziehungsweise durch eine gute Kombination zu untersuchender Merkmale des Schriftbildes und durch umfangreiches Training anhand vieler Beispiele begegnet werden. Eine weitere zentrale Problemstellung wird im folgenden Abschnitt vorgestellt.

Eine handgeschriebene Zahl kann aus technischen Gründen nicht als Ganzes erkannt werden. Die Zahl muss deshalb in einzelne Ziffern zerlegt (segmentiert) werden. Bei der nachfolgenden Erkennung der Ziffern führt ein Fehler der Segmentierung in den meisten Fällen zu einem Erkennungsfehler der betroffenen Ziffer(n) und somit auch der Zahl. Diese Problematik kann sogar als Dilemma bezeichnet werden: Die korrekte Segmentierung einer Zahl ist oftmals ohne die Kenntnis der darin enthaltenen Ziffern nicht möglich, und die korrekte Erkennung der darin enthaltenen Ziffern kann erst nach richtiger Segmentierung erfolgen.

Abbildung 2.1: Beispiel eines handgeschriebenen Wortes

Abbildung 2.1 soll dieses Dilemma anhand des Wortes 'minimum' veranschaulichen. Eine zuverlässige Segmentierung dieses Wortes in seine Buchstaben ist ohne Kenntnis seines "Wahrheitswertes" kaum möglich. Die Problematik der Segmentierung wird in Abschnitt 5.1 bei der Erkennung von Ziffernpaaren konkreter behandelt, und in Abschnitt 6.2 werden verschiedene Ansätze zur Segmentierung von Zahlen vorgestellt.

2.2 Einordnung und Teilgebiete

Da sich die maschinelle Handschrifterkennung mit der Nachbildung menschlicher Fähigkeiten beschäftigt, stellt sie ein Teilgebiet der Künstlichen Intelligenz dar. Wie es für alle Gebiete der Künstlichen Intelligenz zutreffend ist, muss die Aufgabenstellung auch bei der Handschrifterkennung stark eingeschränkt werden.

Die Folgen solcher Einschränkungen liegen in einer weiteren Aufspaltung der Handschrifterkennung in Teilprobleme. Folgende Kriterien spielen bei der Aufteilung eine Rolle: Das Vorhandensein von Informationen zum zeitlichen Verlauf der Erstellung einer Schriftprobe, Einschränkung der zugelassenen Wörter oder Zeichen, Auflagen an die Handschrift selbst (z.B. duch ein Musteralphabet), Auflagen zur Lokalisierung wie das Vorgeben von Kästchen, in die geschrieben werden muss usw.

In den folgenden Absätzen wird die Aufteilung der Handschrifterkennung in *on-line* und *off-line* Erkennung sowie die Unterscheidung der Erkennung ganzer Zahlen und Wörter von der Erkennung isolierter Zeichen behandelt.

Von on-line Erkennung wird gesprochen, wenn die Handschrift in Form zeitlich geordneter Koordinaten vorliegt. Wenn keine zeitliche Information zur Handschrift verfügbar ist, spricht man von off-line Erkennung.

Durch das Weglassen der Zeitinformationen können on-line Daten in off-line Daten transformiert werden. Die umgekehrte Transformation ist sehr aufwendig, schwierig und manchmal sogar unmöglich[1]. Durch die zusätzlichen Zeitinformationen ist eine on-line Erkennung in der Regel besser und zuverlässiger als die off-line Erkennung von Handschrift. In vielen praktischen Anwendungen ist es jedoch nicht möglich, Zeitinformationen zur Erkennung beizuziehen, da handschriftliche Informationen meist erst einige Zeit nach ihrer Entstehung durch einen Scanner erfasst werden.

Wie bereits in Abschnitt 2.1 angesprochen, ist das Problem der Segmentierung sehr komplex. In vielen Fällen wird deshalb versucht, die Segmentierung zu vermeiden oder erheblich zu vereinfachen. Dies kann durch die Vorgabe von Kästchen erreicht werden, bedingt jedoch, dass sich die Benutzer entsprechender Formulare daran halten, ausschliesslich in die Kästchen zu schreiben und pro Kästchen nur zugelassene Zeichen zu verwenden. In solchen Fällen kann sich die Handschrifterkennung auf isolierte Zeichen beschränken. Abbildung 2.2 zeigt dazu einige Besipiele isolierter Zeichen.

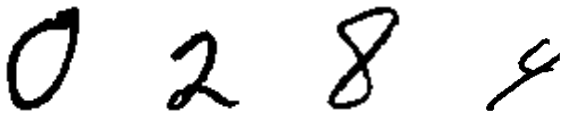


Abbildung 2.2: Beispiele handgeschriebener, isolierte Ziffern

Bei der Erfassung von Dokumenten, die keine Vorgaben zur Separierung der einzelnen Zeichen machen, hat auch die Segmentierung durch die Handschrifterkennung zu erfolgen.

Abbildung 2.3: Beispiele handgeschriebener Zahlen

Wie dies durch die Zahlen in Abbildung 2.3 illustriert wird, müssen auch zusammenhängende Ziffern oder Ziffern, die aus mehreren Teilstücken bestehen, korrekt segmentiert werden können.

[1]Roger Ammann hat sich in seiner Diplomarbeit mit dieser Problematik beschäftigt [Amm96].

2.3 Anwendungen

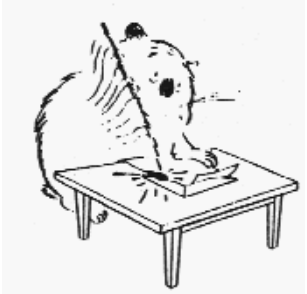
In diesem Abschnitt werden exemplarisch zwei Anwendungen der Handschrifterkennung vorgestellt. Eine Anwendung lässt sich in "intelligenten" Notitzblöcken oder PDA's (Personal Digital Assistants) finden. Eine andere Anwendung verwendet die Handschrifterkennung zur maschinellen Verarbeitung von Betragsfeldern auf Einzahlungsscheinen.

PDA's oder "intelligente" Notitzblöcke verfügen über einen speziellen, berührungsempfindlichen Bildschirm. Mit einem Stift gemachte Eingaben können dadurch fortlaufend digitalisiert werden. Mit dar Erfassen der Stiftbewegung fallen somit Positions- und Zeitinformationen an, die eine on-line Erkennung der Handschrift des Benutzers ermöglichen. Um eine brauchbare Zuverlässigkeit zu erreichen, müssen diese Geräte von ihren Besitzerinnen oder Besitzern zuerst mit deren persönlichen Handschrift trainiert werden.

Die Interpretation von Betragsfeldern bei Einzahlungsscheinen stellt eine Anwendungsmöglichkeit der off-line Handschrifterkennung dar, da Einzahlungsscheine immer vor deren Digitalisierung ausgefüllt werden. Ein Beispiel eines von Hand ausgefüllten Betragsfeldes gibt Abbildung 2.4.

Abbildung 2.4: Beispiel eines Betragfeldes eines Einzahlungsscheines

Da Sicherheitsanforderungen für den Zahlungsverkehr eine entscheidende Rolle spielen, werden von Hand ausgefüllte Einzahlungsscheine von zwei verschiedenen Personen erfasst. Falls die Interpretationen der beiden Personen nich übereinstimmen, wird der Einzahlungsschein ein drittes Mal erfasst, um die korrekte Erkennung sicherzustellen. Dies gilt sowohl für die dezentrale Erfassung an Postschaltern, als auch für die zentrale Erfassung von Zahlungsaufträgen. Durch die Anwendung maschineller Handschrifterkennung könnte beispielsweise eine der beiden Erfassungen eines Betragfeldes durch den Computer erfolgen.



Kapitel 3

Bilddaten

Um Systeme zur Erkennung von Handschrift trainieren und testen zu können, werden umfangreiche Sammlungen von Beispielen handgeschriebener Ziffern, Zahlen, Buchstaben usw. benötigt. Jedes Beispiel besteht dabei aus einem Bild und einem zugehörigen Wahrheitswert, der die korrekte Interpretation des Bildinhaltes wiedergibt. Ein Erkennungssystem hat somit ein Beispiel genau dann als richtig erkannt, wenn das Resultat des Erkennungsvorganges mit dem Wahrheitswert des Beispieles übereinstimmt.

Das Erfassen und Prüfen dieser Beispieldaten ist jedoch sehr aufwendig. Damit diese Arbeiten nicht immer wieder von neuem durchgeführt werden müssen, werden Bilddaten zur Handschrifterkennung häufig in speziellen Datenbanken gespeichert.

Dieses Kapitel geht in Abschnitt 3.1 auf die wichtigsten Anforderungen an solche Datenbanken ein. Abschnitt 3.2 behandelt das Testen von Erkennungssystemen mit vorgegebenen Beispieldaten und Abschnitt 3.3 stellt diejenigen Datenbanken vor, die in dieser Arbeit für das Training und die Tests der Erkenner verwendet wurden.

3.1 Anforderungen an Datenbanken

Die Problematik der Handschrifterkennung ist sehr stark vom Umfeld abhängig, in dem ein Erkennungssystem eingesetzt werden soll. Hohe Erkennungsraten lassen sich nur dann erreichen, wenn für das Training und die Tests Beispieldaten verwendet werden, die aus dem vorgesehenen Umfeld stammen. Ein System zur Erkennung von Postleitzahlen sollte deshalb auch mit Bildern aus dem Postverkehr trainiert beziehungsweise getestet werden.

Eine erste, zentrale Anforderung an eine Datenbank zur Handschifterkennung besteht also darin, dass die darin enthaltenen Beispiele für die geplante Anwendung

repräsentativ sind. Bei der Erfassung der Beispieldaten ist ausserdem darauf zu achten, dass dieselbe Techniken und Verfahren eingesetzt werden, wie sie für den praktischen Einsatz des Handschrifterkennungssystems vorgesehen sind.

Falls Erkennungsraten verschiedener Handschrifterkennungssysteme anhand publizierter Messungen miteinander verglichen werden sollen[1], ist es unerlässlich, dass die beiden Systeme mit derselben Sammlung von Beispielen getestet wurden. Um solche Vergleiche zu ermöglichen, müssen die verwendeten Datenbanken somit öffentlich zugänglich sein. Identische Testbedingungen können dann am einfachsten erreicht werden, wenn die Testdaten explizit als solche gekennzeichnet sind. Dadurch wird auch festgelegt, welche Beispiele nicht zum Training oder zur Optimierung von Erkennungssystemen verwendet werden dürfen.

Die unterschiedlichen Anforderungen an Trainings- und Testdaten, werden in den folgenden beiden Abschnitten behandelt.

3.1.1 Anforderungen an Trainingsdaten

Die Anforderungen an Trainingsdaten ergeben sich aus dem Wunsch nach guten Erkennungsraten und hoher Zuverlässigkeit der damit zu trainierenden Erkennungssysteme. Die Trainingsdaten müssen vollständig, korrekt und im erforderlichen Umfang vorhanden sein.

Vollständigkeit lässt sich nur durch eine breite Abdeckung der unterschiedlichen Fälle der Praxis erreichen. Um Verfahren zur Erkennung von Spezialfällen entwickeln zu können, müssen auch solche Beispiele in genügender Anzahl vertreten sein.

Die Korrektheit der Beispiele (Übereinstimmung von Bildinhalt und Wahrheitswert) muss in der Regel manuell durch einen entsprechenden Vergleich überprüft werden[2]. Durch die Verwendung korrekter Beispiele wird sichergestellt, dass ein Erkennungssystem nicht mit falschen Beispielen trainiert wird.

Die Zahl der Beispiele, die zum Training eines Erkennungssystems benötigt werden, ist von den eingesetzten Verfahren abhängig. Zum Training eines Erkennungssystems für handgeschriebenen Ziffern können bereits mehrere 10000 Beispiele erforderlich sein[3].

[1] Eines der Ziele dieser Diplomarbeit besteht darin, das implementierte Erkennungssystem mit bereits publizierten Forschungsarbeiten zu vergleichen (siehe Abschnitt 1.1).

[2] Die Zuordnung der Wahrheitswerte und deren Prüfung stellt meistens den aufwendigsten Teil bei der Erstellung einer Datenbank zur Handschrifterkennung dar.

[3] Der Zusammenhang zwischen der Anzahl der verwendeten Trainingsbeispielen und der Erkennungsrate wird in [Kre91] im Detail besprochen.

3.1.2 Anforderungen an Testdaten

Testdaten bilden die Grundlage zur Messung von Erkennungsraten. Um zuverlässige Aussagen über die Leistungsfähigkeit eines Erkennungssystems machen zu können, müssen für die Testdaten die Kriterien Repräsentativität, Korrektheit und genügender Umfang erfüllt werden.

Die selektierten Testbeispiele müssen für die gewählte Anwendung repräsentativ sein, im Gegensatz zu den Trainingsdaten sollten Spezialfälle deshalb nicht überproportional häufig vertreten sein.

Da ein Beispiel genau dann als richtig erkannt gilt, wenn das Resultat des Erkennungssystems mit dem Wahrheitswert des Bildes übereinstimmt, verfälschen unkorrekte Wahrheitswerte die Messungen.

Die letzte Anforderung betrifft den Umfang der Testdaten. Die Anzahl der Beispiele die für Tests notwendig sind, ist nicht von den zur Erkennung eingesetzten Verfahren abhängig, sie lässt sich allein durch die geforderte Messgenauigkeit bestimmen. Der folgende Abschnitt behandelt diese Thematik eingehender.

3.2 Testen von Erkennungssystemen

Erkennungssysteme werden getestet, um Aussagen über ihre Leistungsfähigkeit machen zu können. Die Leistungsfähigkeit wird in der Regel durch Messung der Erkennungsrate, der Fehlerrate und der Rückweisungsrate experimentell bestimmt.

In der vorliegenden Diplomarbeit werden dazu die untenstehenden Definitionen verwendet[4], wobei N der Anzahl der im Experiment verwendeter Beispiele entspricht, N_{cor} der Anzahl richtig erkannter Beispiele, N_{err} der Anzahl nicht erkannter Beispiele und N_{rej} der Anzahl der zurückgewiesenen Beispiele entspricht. Mit $N_{cor} + N_{err} + N_{rej} = N$ ergibt sich somit $R_{cor} + R_{err} + R_{rej} = 1$.

$$Erkennungsrate\ R_{cor} = \frac{N_{cor}}{N} \tag{3.1}$$

$$Fehlerrate\ R_{err} = \frac{N_{err}}{N} \tag{3.2}$$

$$Rückweisungsrate\ R_{rej} = \frac{N_{rej}}{N} \tag{3.3}$$

Wie bereits erwähnt, gilt ein Beispiel genau dann als richtig erkannt, wenn das

[4]Die Definitionen wurden von C.K. Chow in [Cho57] und [Cho70] vorgeschlagen.

Resultat des Erkennungssystems und der Wahrheitswert des Beispieles identisch sind. Falls das Erkennungssystem zu einem Beispiel kein Resultat ausgibt, gilt dieses Beispiel als zurückgewiesen. Falls das Erkennungssystem ein Resultat ausgibt und dieses nicht mit dem Wahrheitswert des Beispieles übereinstimmt, wird das Beispiel als nicht erkannt bewertet.

Für viele Anwendungsgebiete der Handschrifterkennung spielt die Fehlerrate von Erkennungssystemen eine zentrale Rolle. Man ist deshalb daran interessiert, die Fehlerrate experimentell zuverlässig abschätzen zu können. Nach Abschnitt 10.3 in [Dev82] kann die erreichte Genauigkeit der Fehlerrate durch die Angabe eines Vertrauensintervalles berechnet werden[5].

Die tatsächliche Fehlerrate E eines Erkennungssystem liegt für $z = 1.96$ mit einer Wahrscheinlichkeit von 95% im folgenden Vertrauensintervall:

$$[R_{err} - z\hat{\sigma}(R_{err}), R_{err} + z\hat{\sigma}(R_{err})] \tag{3.4}$$

Wobei mit

$$\hat{\sigma}(R_{err}) = \sqrt{\frac{R_{err}(1 - R_{err})}{N}} \tag{3.5}$$

die geschätzte Standardabweichung von R_{err} bezeichnet wird. Ein kleines Beispiel soll diese Berechnungen illustrieren.

In einem Experiment hat ein gegebenes Erkennungssystem 400 von insgesamt 500 Beispielen richtig und 12 falsch erkannt. Die restlichen 88 Beispiele wurden vom Erkennungssystem zurückgewiesen. Die gemessene Fehlerrate R_{err} beträgt 2,4% mit einer geschätzten Standardabweichung $\hat{\sigma}(R_{err})$ von 0.68%. Somit liegt die effektive Fehlerrate E mit einer Wahrscheinlichkeit von 95% zwischen 1.72% und 3.08%.

3.3 Verwendete Datenbanken

In diesem Abschnitt werden die beiden Datenbanken vorgestellt, die in dieser Diplomarbeit zum Trainieren und Testen verwendet wurden. Dabei handelt es sich um zwei amerikanische Datenbanken, die erst vor einigen Jahren veröffentlicht wurden und somit weltweit verfügbar sind.

[5] Dabei wird vorausgesetzt, dass repräsentative, korrekte Beispiele zur Messung der Fehlerrate verwendet werden, die weder zum Training noch zur Optimierung des Erkennungssystems zum Einsatz kommen.

3.3.1 CEDAR Datenbank

Die CEDAR-Datenbank entstand durch eine Zusammenarbeit von *United States Postal Services* (USPS) und *Center of Excellence for Document Analysis and Recognition* (CEDAR) [Hul94]. Sie enthält Bilder von handgeschriebenen Ziffern, Zahlen, Buchstaben und Wörtern, welche den erfassten Adressen entnommen wurden. Zur Erfassung der Adressen wurden 1987 und 1988 in der Hauptpoststelle in Buffalo (New York) Briefe mit handgeschriebenen Adressen ausgewählt. Mit einem Scanner wurden die Adressen der selektierten Briefe bei einer Auflösung von 300 dpi[6] mit 256 Graustufen digitalisiert.

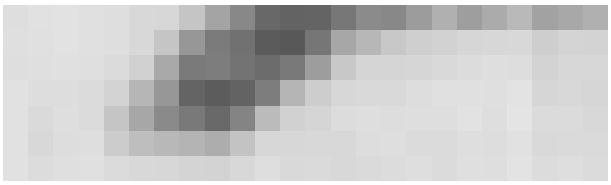


Abbildung 3.1: Vergrösserter Ausschnitt einer Postleitzahl

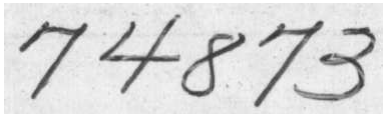
Abbildung 3.1 stellt einen vergrösserten Ausschnitt einer digitalisierten Postleitzahl dar, um deren Bildpunktstruktur zu visualisieren. Dabei handelt es sich um den Anfang des oberen Bogens der Ziffer '2' aus Abbildung 1.1.

Nach der Erfassung der Adressen wurden diese von Hand in die Teilbilder 'Stadt', 'Staat' und 'Postleitzahl' segmentiert und den so gewonnenen Beispieldaten für Worte und Zahlen deren Wahrheitswerte zugeordnet. Abbildung 3.2 zeigt einige Beispiele extrahierter Postleitzahlen.

Abbildung 3.2: Beispiele für Zahlen der CEDAR Datenbank

Etwa zehn Prozent dieser Bilder wurden zufällig ausgewählt und als Testdaten markiert, wodurch deren Repräsentativität sichergestellt sein sollte. Um ausserdem Trainings- und Testdaten für die Buchstaben- und Ziffernerkennung zur Verfügung

[6] Mit der Einheit dpi (Dots Per Inch) wird die Anzahl der Bildpunkte pro Längeneinheit gemessen.



(a)

(b) (c) (d) (e) (f)

Abbildung 3.3: Beispiel einer segmentierten Zahl (Datei bd0010.0)

zu stellen, wurde ein Teil der Beispiele weiter in die Buchstaben sowie Ziffern segmentiert und mit Wahrheitswerten versehen. Abbildungen 3.3 zeigt eine in Ziffern segmentierte Zahl der CEDAR Datenbank.

Die aus der Zahl segmentierten Ziffern sind dabei in einzelnen Dateien gespeichert worden. Die Bilder der Abbildungen 3.3(b), (c), (d), (e) und (f) tragen die folgenden Dateinamen: 'bd0010_7.18', 'bd0010_4.19', 'bd0010_8.20', 'bd0010_7.26' und 'bd0010_3.27', wobei der Wahrheitswert der Ziffern direkt vor dem Punkt des Dateinamens steht.

Dass die Segmentierung der Zahlen in die Ziffern bei der CEDAR Datenbank nicht immer korrekt ist, belegt das Beispiel in Abbildung 3.4.

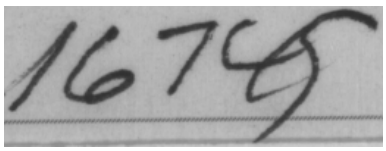
Eine Übersicht der Organisation der Datenbank ist in Tabelle 3.1 wiedergegeben. In den Teilmengen 'bindigis' und 'binzips' sind im Gegensatz zur Teilmenge 'zipcodes' alle Bilder binärisiert[7].

Teilmenge	Training		Test	
	Zahlen	Ziffern	Zahlen	Ziffern
bindigis	-	18468	-	4924
binzips	-	-	495	-
zipcodes	9019	-	435	-

Tabelle 3.1: Organisation der CEDAR Datenbank

Die weitere Aufteilung der Teilmengen 'bindigis', 'binzips' und 'zipcodes' erfolgt

[7]Die Bilder enthalten nur noch schwarze und weisse Bildpunkte.



(a)

(b) (c) (d) (e) (f)

Abbildung 3.4: Falsch segmentierte Zahl (Datei bs0133.0)

nach der Art der Selektion der Beispieldaten in die Untermengen 'bd', 'bl', 'bc', 'bs', 'bb', 'br' und 'bu'. Weiterführende Informationen zur CEDAR Datenbank sind zusammen mit allen Bilddaten auf der veröffentlichten CD-ROM vorhanden.

Die CEDAR Datenbank stellt durch ihre Realitätsnähe eine echte Herausforderung für Erkennungssysteme dar. Ihre explizite Aufteilung in Trainings- und Testdaten ermöglicht zudem den direkten Vergleich von Erkennungssystemen, die mit den CEDAR Daten trainiert und getestet wurden.

3.3.2 NIST SD3 Datenbank

In dieser Diplomarbeit wurde auch die im Jahre 1992 veröffentlichte NIST SD3 Datenbank verwendet. Diese Datenbank wurde vom Amerikanischen *National Institute of Standards and Technology* im Rahmen einer Konferenz [WGJ+92] zur Bestimmung des 'state-of-the-art' der Erkennung isolierter Buchstaben und Ziffern den Konferenzteilnehmern zur Verfügung gestellt.

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

Abbildung 3.5: Aus Formularen extrahierte Zahlen der NIST SD3 Datenbank

Teilmenge	Training		Test	
	Zahlen	Ziffern	Zahlen	Ziffern
f0000 - f1799	45000	216000	-	-
f1800 - f1899	-	-	2500	12000
f1900 - f1999	2500	12000	-	-
f2000 - f2099	-	-	2500	12000

Tabelle 3.2: Aufteilung der NIST SD3 Datenbank in Trainings- und Testdaten

In der SD3 Datenbank sind die Bilder 2100 digitalisierter Formulare gespeichert, die von Beamten der Amerikanischen Volkszählungsbehörde ausgefüllt worden sind. Die Formulare sind speziell für die oben genannte Konferenz konzipiert worden und enthalten unter anderem 25 Felder für zwei- bis sechsstellige Zahlen. Zur Erfassung der Formulare wurden diese bei einer Auflösung von 300 dpi digitalisiert und anschliessend binärisiert. Da die Zahlen in der SD3 Datenbank nicht direkt zur Verfügung gestellt werden, mussten die entsprechen Felder aus den Formularen extrahiert werden (siehe Abbildung 3.5 für einige Beispiele). Dazu wurden Teile einer entsprechenden Software verwendet, deren Source-Code auf der *hsfsys* CD-ROM durch NIST veröffentlicht wurde.

Die Extraktion der Zahlen aus den entsprechenden Feldern der Formulare ist jedoch nicht sehr stabil. Beispielsweise beträgt der Wahrheitswertes der Abbildung 3.5(d) '12791', was bedeutet, dass die letzte Ziffer beim Extrahieren des Feldes abgeschnitten wurde.

Die Bilddaten der NIST SD3 Datenbank sind nicht in Trainings- und Testdaten aufgeteilt, da den Konferenzteilnehmern alle Bilder der SD3 Datenbank zur Entwicklung und zum Training ihrer Erkennungssysteme zur Verfügung gestellt wurden. Die an der Konferenz verwendeten Testdaten wurden durch NIST mit der Datenbank SD7 publiziert. Leider erfolgte die Sammlung der Beispiele für SD3 und SD7 nicht im selben Umfeld. Die SD7 enthält nur Daten von Formularen, die durch Studenten einer High School ausgefüllt wurden. Dadurch erreichten die Erkennungssysteme, die mit Daten der SD3 trainiert wurden auf den Daten der SD7 keine optimalen Ergebnisse [WGJ+92].

Ein weiterer Nachteil der NIST SD7 besteht darin, dass sie nur noch die aus den Formularen extrahierten Ziffern und Buchstaben enthält. Die SD7 stellt somit auch keine Testdaten für Systeme zur Erkennung von Zahlen bereit, was zur Folge hat, dass für den Fall der Zahlenerkennung die SD3 in Trainings- und Testdaten aufgeteilt werden muss. Die in dieser Diplomarbeit verwendetete Aufteilung der SD3 stützt sich auf die von [KR92] verwendete Aufteilung und ist in Tabelle 3.2 dokumentiert.

Die in dieser Tabelle angegebenen Zahlen für die Anzahl vorhandener Beispiele sind

Schätzungen. Für jedes Formular wurde mit einer "Ausbeute" von 25 Zahlen und 120 Ziffern gerechnet. Die Teilmengen sind durch die angegebenen Formularnamen spezifiziert, wobei der Formularname 'f0214' beispielsweise zum 214. Formular gehört.

Die NIST SD3 Datenbank ist gegenüber der CEDAR Datenbank in dem Sinne unnatürlich, dass den Testpersonen bewusst war, dass die Formulare zur maschinellen Erkennung konzipiert worden sind und sie vorgegebene Zahlen in vorgegebene Felder zu schreiben hatten. Diese zusätzlichen Einschränkungen führen dazu, dass die NIST Daten maschinell leichter zu erkennen sind als die CEDAR Daten.

Kapitel 4

Ein Ziffernerkenner

In diesem Kapitel wird ein Erkennungssystem für handgeschriebene Ziffern kurz vorgestellt, das in [Gro94] genauer beschrieben wurde. Das Kapitel soll einen Einblick in die Ziffernerkennung vermitteln und einige konkrete Methoden zur Erkennung von Ziffern erklären. Einige dieser Methoden wurden auch zum Bau des Paarerkenners verwendet, der im nächsten Kapitel beschrieben wird.

Abschnitt 4.1 beschäftigt sich mit der Aufgabenstellung und Abschnitt 4.2 mit dem Aufbau eines Ziffernerkenners.

4.1 Aufgabenstellung

Der Ziffernerkenner soll für ein gegebenes Bild einer isolierten Ziffer als Resultat die optimale Interpretation aus den Ziffernklassen '0', '1', ..., '9' bestimmen und eine Wertung $\in [0,1]$ berechnen, die ein Mass für die Korrektheit des Resultates darstellt.

Abbildung 4.1: Schema eines Ziffernerkenners

Das Schema eines solchen Ziffernerkenners wird durch obige Abbildung veranschaulicht. Dabei werden Daten als Ellipsen und Prozesse als Rechtecke dargestellt. Schat-

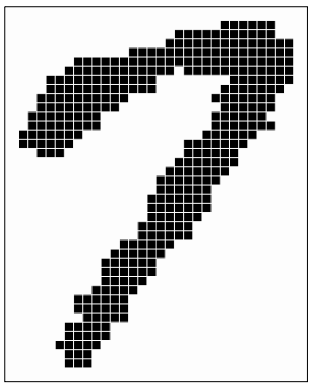


Abbildung 4.2: Beispiel einer Eingabe für den Ziffernerkenner

tierte Rechtecke kennzeichnen Prozessgruppen, die in dieser Dokumentation noch weiter verfeinert werden. Für den Ziffernerkenner stellt Abbildung 4.3 die weitere Verfeinerung der Abbildung 4.1 dar.

Es ist zu beachten, dass in der Aufgabenstellung vorausgesetzt wird, dass es sich beim Eingabebild um eine isolierte Ziffer handelt. Der Ziffernerkenner soll nicht entscheiden müssen, ob es sich bei der Eingabe überhaupt um eine Ziffer handelt.

Auf die Eingabe eines Bildes reagiert die Ziffernerkennung gemäss Aufgabenstellung mit der Ausgabe des Resultates und der zugehörigen Wertung. Ein Beispiel für ein Bild wird durch Abbildung 4.2 gegeben. Die zugehörigen Ausgaben des Ziffernerkenners lauten für das Resultat '7' und die Wertung 0.998.

Mit der hohen Wertung von 0.998 gibt der Ziffernerkenner zu verstehen, dass er sich sehr "sicher" ist, dass es sich bei Abbildung 4.2 um eine '7' handelt.

4.2 Aufbau

Der Aufbau des Ziffernerkenners, der hier vorgestellt wird, ist genereller Natur und kann für verschiedene Erkennungsaufgaben verwendet werden. Er basiert auf den drei Modulen 'Vorverarbeitung', 'Merkmalsextraktion' und 'Klassifikation'. In Abbildung 4.3 ist die Verknüpfung der grundlegenden Module des Ziffernerkenners graphisch dargestellt.

In der Vorverarbeitung wird das Bild zuerst in ein Zwischenbild B_{norm} transformiert, dann werden in der Merkmalsextraktion für Ziffern charakteristische Merkmale von B_{norm} bestimmt und zu einem Merkmalsvektor MV zusammengefasst. In der Klassifikation wird der so gewonnene Merkmalsvektor mit der Wissensbasis

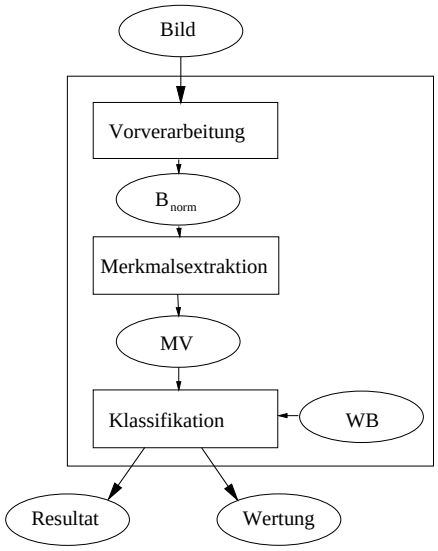


Abbildung 4.3: Aufbau eines Ziffernerkenners

WB des Klassifikators verglichen. Aufgrund dieses Vergleiches kann schliesslich die wahrscheinlichste Bedeutung des Merkmalsvektors bestimmt werden.

In den folgenden drei Unterabschnitten werden die Module Vorverarbeitung, Merkmalsextraktion und Klassifikation detaillierter beschrieben.

4.2.1 Vorverarbeitung

Die Idee der Vorverarbeitung besteht einerseits darin, die "Datenflut" des Eingabebildes zu verringern, ohne dadurch problemrelevante Informationen zu verlieren und andererseits in einer Normierung des zu erkennenden Bildes. Dabei entsteht ein Zwischenbild, dessen nachfolgende Merkmalsextraktion durch die bei der Vorverarbeitung erfolgte Normierung vereinfacht wird. Da die optimale Vorverarbeitung sehr stark von der konkreten Problemstellung abhängig ist, können keine allgemeinen Verfahren angegeben werden. Bei der Wahl der Verfahren sind deren Einfachheit und Zweckmässigkeit entscheidend, wobei sich diese Kriterien häufig erst durch praktische Messungen verifizieren lassen.

Für diesen Ziffernerkenner besteht die Vorverarbeitung aus einer Positions- und einer Grössennormierung. Zuerst wird im Eingabebild das kleinste Rechteck bestimmt, das alle schwarzen Bildpunkte enthält[1], dann wird dieser Bildausschnitt auf eine vorgegebene Breite und Höhe transformiert.

[1] Dieses Rechteck wird auch als 'Bounding Box' bezeichnet.

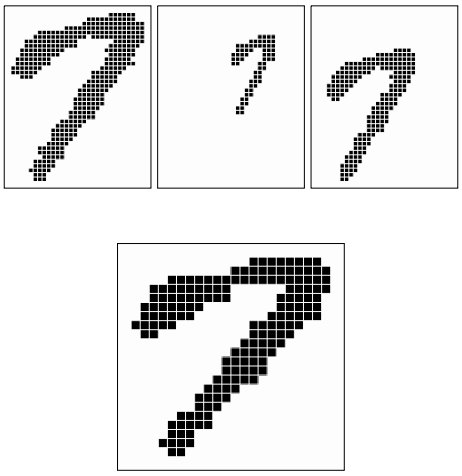


Abbildung 4.4: Drei verschiedene Eingabebilder und das resultierende Zwischenbild B_{norm} der Vorverarbeitung

Abbildung 4.4 demonstriert, wie durch dieses Vorgehen unterschiedliche Eingabebilder in dasselbe normierte Zwischenbild (der Grösse 24×24 Bildpunkte) transformiert werden können. Ausserdem lässt sich durch diese Vorverarbeitung die nicht relevante Information der Position und Grösse der Ziffer unterdrücken.

4.2.2 Merkmalsextraktion

Die Merkmalsextraktion stellt eine Konzentration der Information der Bilddaten dar. Man versucht dabei, ausschliesslich Merkmale aus dem Bild zu extrahieren, welche für die zu unterscheidenden Ziffernklassen charakteristisch sind. Die pro Bild extrahierten Merkmale werden dann zu einem Merkmalsvektor (MV) zusammengesetzt.

Die Merkmale, die für den hier vorgestellten Ziffernerkenner verwendet werden, basieren auf den Konturen, die aus dem Zwischenbild B_{norm} extrahiert werden. Neben konturbasierten Merkmalen haben sich viele weitere Merkmale in der Praxis bewährt, die unterschiedliche Vor- und Nachteile aufweisen. In den folgenden Absätze wird die Bestimmung der konurbasierten Merkmale im Überblick beschrieben.

In einem ersten Schritt werden die Konturen einer Ziffer zusammen mit ihrer Richtung bestimmt, was durch die Anwendung von Sobel-Operatoren[2] realisiert werden

[2] Das Kapitel 'Konturliniendetektion' des Skriptums zur Vorlesung 'Bildanalyse' behandelt diese Thema detaillierter [Bun95].

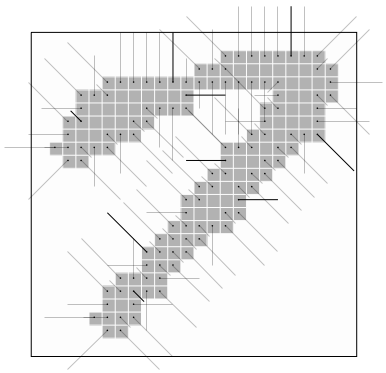


Abbildung 4.5: Aus dem Zwischenbild B_{norm} extrahiertes Konturbild

kann. Dabei wird für jeden Bildpunkt (i,j) die vorherrschende Richtung bestimmt und quantisiert. Es werden acht Richtungen unterschieden, deren Bezeichnung im linken Schema der Abbildung 4.6 angegeben ist. Abbildung 4.5 stellt ein so berechnetes Konturbild dar.

In einem zweiten Schritt wird das resultierende Konturbild in 3×3 Regionen aufgeteilt (rechtes Schema in Abbildung 4.6), und für jede Region ein Richtungshistogramm erstellt.

In Abbildung 4.7 ist die Aufteilung des Konturbildes in 3×3 Regionen zusammen mit den berechneten Konturhistogrammen dargestellt. Die Region unten rechts enthält keine Bildpunkte, die zur Kontur der Ziffer gehören und somit weist auch das zugehörige Histogramm für fast alle Konturrichtungen nur sehr kleine Werte auf.

Zur Bildung des Histogrammes werden die quantisierten Richtungen aller Kontourpunkte über die Summationsfläche einer Region entsprechend gewichtet und zusammengezählt. Die Summationsfäche einer Region reicht dabei von der Mitte der entsprechenden Region bis zu den Mitten der Nachbarregionen wie dies im rechten Schema der Abbildung 4.6 für die Region 1 dargestellt ist.

Das zu bestimmende Richtungshistogramm setzt sich aus Werten $h_{k,l}$ zusammen, wobei $k \in \{1, 2, ..., 9\}$ die gewünschte Region und $l \in \{0, 1, ..., 7\}$ die gewählte Richtung angibt. Die Werte $h_{k,l}$ sind durch die folgende Funktion definiert

$$h_{k,l} = \sum_{i,j}^{SF_k} r_l(i,j) \cdot g(di, dj) \tag{4.1}$$

Der Richtungswert für die Richtung l im Bildpunkt (i, j) wird mit $r_l(i, j)$ bezeichnet und $g(di, dj)$ enspricht der Gewichtung des entsprechenden Bildpunktes. Dieser be-

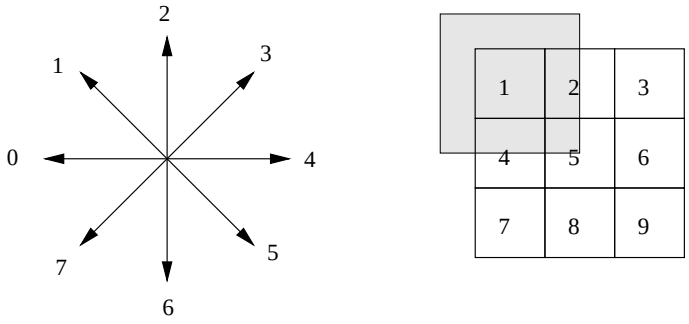


Abbildung 4.6: Bezeichnung der Richtungen und Regionen zur Histogrammbildung

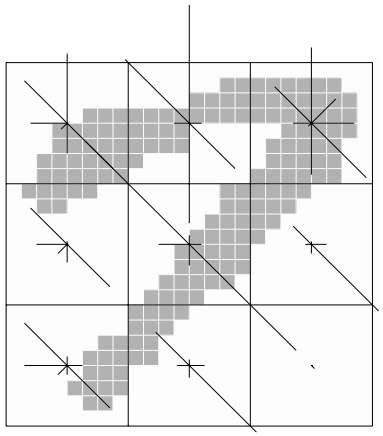


Abbildung 4.7: Aufteilung des Konturbildes in Regionen mit den zugehörigen Konturhistogrammen

findet sich im Abstand von di Bildpunkten vertikal und dj Bildpunkten horizontal von der Mitte der Summationsfläche SF_k, deren Breite und Höhe mit b beziehungsweise mit h bezeichnet ist. Die Gewichtung $g(di, dj)$ der Richtungswerte erfolgt durch untenstehende Funktion.

$$g(di, dj) = cos(di\frac{\pi}{2h}) \cdot cos(dj\frac{\pi}{2b}) \tag{4.2}$$

Die berechneten Histogrammwerte $h_{k,l}$ werden nach folgendem Schema zum Merkmalsvektor MV zusammengesetzt:

$$MV = (h_{1,0}, h_{1,1}, \ldots, h_{1,7}, h_{2,0}, \ldots, h_{9,7}) \tag{4.3}$$

Aus Abbildung 4.7 geht somit der Merkmalsvektor MV hervor, der in Abbildung 4.8

(24, 41, 46, 0, 1, 45, 21, 6,
10, 61, 78, 0, 8, 43, 67, 1,
13, 34, 50, 24, 28, 52, 34, 3,
21, 35, 2, 0, 1, 42, 13, 9,
20, 97, 6, 0, 9, 101, 14, 0,
5, 18, 3, 0, 10, 64, 6, 0,
29, 42, 6, 0, 11, 41, 6, 11,
9, 31, 4, 0, 11, 63, 9, 2,
0, 0, 0, 0, 0, 3, 0, 0)

Abbildung 4.8: Aus den Konturhistogrammen gewonnener Merkmalsvektor

dargestellt ist. Da jedes Histogramm acht Richtungen unterscheidet, ergibt sich für den MV die Dimension 72. Jede Zeile in Abbildung 4.8 stellt dabei ein Histogramm einer Region dar. Die erste Zeile enthält das Histogramm für die Region 1, die zweite Zeile das Histogramm für die Region 2, usw.

Wie im folgenden Abschnitt detaillierter dargestellt wird, spielen die Merkmalsvektoren sowohl für das Training des Ziffernerkenners wie auch für die eigentliche Erkennung von Ziffern eine zentrale Rolle.

4.2.3 Klassifikation

In der Klassifikation wird anhand eines extrahierten Merkmalsvektors und der Wissensbasis des Ziffernerkenners entschieden, welche Ziffernklasse aus '0', '1', ...,'9' für das zu erkennende Beispiel die optimale Interpretation darstellt. Neben dieser Interpretation kann aus dem Vorgang der Klassifkation auch ein Mass für deren Korrektheit abgeleitet werden.

Als mögliches Verfahren zur Klassifikation soll hier das Verfahren des 'Nearest Neighbor' (oder Nächsten Nachbarn, abgekürzt NN) besprochen werden. Neben dem Nearest Neighbour und ihm verwandten Verfahren, existieren auch andere Methoden zur Klassifikation von Merkmalsvektoren wie Neuronale Netzwerke oder Polynomiale Klassifikatoren.

Durch die Wahl des Klassifikators wird gleichzeitig auch die Repräsentation des "Wissens" in der Wissensbasis eines Erkennungssystems festgelegt. Im Falle eines NN-Klassifikators besteht die Wissensbasis aus einer Menge von Prototypen pro Klasse, welche aus den zum Training bestimmten Bilddaten extrahiert wurden. Die Prototypen bestehen dabei aus einem Merkmalsvektor und einem zugehörigen Wahrheitswert.

Das Training eines NN-Klassifikators erfolgt nach einem simplen Schema: Jedes Bild,

i	WW	M1	M2	M3	...	M72
...						
1234	'8'	4	46	25		7
1235	'9'	7	36	57		7
1236	'0'	14	25	43		13
1237	'1'	7	3	23		12
1238	'2'	11	19	57		9
...						

Tabelle 4.1: Auszug der Wissensbank des Ziffernerkenners

das trainiert wird, durchläuft den Erkenner bis zur Merkmalsextraktion. Der so gewonnene Merkmalsvektor wird dann zusammen mit dem Wahrheitswert des Bildes zu einem Prototypen zusammengefasst und als Tabelleneintrag in der Wissensbasis gespeichert.

Tabelle 4.1 stellt einen fiktiven Auszug aus der Wissensbasis des Ziffernerkenners dar. Die Spalte 'i' enthält die Nummer des Eintrags der Wissensbasis, die Spalte 'WW' den Wahrheitswert des Prototypen und die Spalten 'M1' bis 'M72' stellen den Merkmalsvektor dieses Prototypen dar.

Um ein Bild zu erkennen, wird zuerst dessen Merkmalsvektor MV bestimmt. Anschliessend wird derjenige Prototyp P_i der Wissensbasis gesucht, welcher den geringsten Abstand d_{min} zum extrahierten Merkmalsvektor MV aufweist[3].

$$d_{min} = \min_i d(MV, P_i) \tag{4.4}$$

Der Abstand d wird hier durch die Euklidsche Distanz zwischen dem Prototypen P_i und dem Merkmalsvektor MV bestimmt, wobei $P_i(j)$ und $MV(j)$ dem j-ten Merkmal des Prototypen beziehungsweise des zu klassifizierenden Merkmalsvektoren entsprechen.

$$d(MV, P_i) = \sqrt{\sum_{j=1}^{72} (MV(j) - P_i(j))^2} \tag{4.5}$$

Falls der Abstand $d_k = d(MV, P_k)$ minimal ist, stellt der zum k-ten Eintrag gehörende Wahrheitswert WW(k) das Resultat der Klassifikation dar. Die Wertung dieses Resultates kann durch folgende Formel bestimmt werden.

$$Wertung = \frac{1}{1 + d(MV, P_k)} \tag{4.6}$$

[3]Dieser Prototyp stellt somit der "nächste Nachbar" von MV in der Wissensbasis dar.

Die beiden folgenden Beispiele sollen zeigen, weshalb diese Wertung als Korrektheitsmass für das Resultat der Klassifikation aufgefasst werden kann.

Eine vollständige Übereinstimmung eines zu klassifizierenden Merkmalsvektors mit einem Prototypen der Wissensbasis ($d_k = 0.0$) führt zu einer Wertung von 1, was beispielsweise dann der Fall ist, wenn dem Ziffernerkenner ein bereits trainiertes Bild vorgelegt wird. Eine Fehlinterpretation ist in diesem Fall sehr unwahrscheinlich.

Falls der Ziffernerkenner ein Bild erkennen soll, das sich sehr stark von allen trainierten Beispielen unterscheidet, wird die Distanz d_{min} gross und die Wertung kommt nahe bei 0.0 zu liegen. Mit dieser tiefen Wertung wird auf eine mögliche Fehlinterpretation hingewiesen.

Kapitel 5

Ein Paarerkenner

In diesem Kapitel wird ein System zur Erkennung handgeschriebener Ziffernpaare vorgestellt, das als Teil der vorliegenden Diplomarbeit entstanden ist. Der Paarerkenner sollte als Teilsystem des Zahlenerkenners eingesetzt werden. Diese Idee musste jedoch aufgrund unbefriedigender Resultate der durchgeführten Experimente wieder verworfen werden.

Zuerst wird die Aufgabenstellung und Motivation für den Paarerkenner im Abschnitt 5.1 behandelt. Abschnitt 5.2 beschäftigt sich mit den Bilddaten, die zum Training und für die Tests des Paarerkenners verwendet wurden. Der Aufbau des Paarerkenners wird im Abschnitt 5.3 vorgestellt. Experimente und Resultate sind im Abschnitt 5.4 dokumentiert und werden im Abschnitt 5.5 diskutiert. Schlussfolgerungen zur Paarerkennung werden im Abschnitt 5.6 gezogen.

5.1 Aufgabenstellung und Motivation

Der Paarerkenner soll für ein gegebenes Bild eines isolierten Ziffernpaares die optimale Interpretation aus den 100 Paarklassen '00', '01', ...,'99' bestimmen und ein Mass für deren Korrektheit berechnen. Bilder von Ziffernpaaren enthalten je zwei Ziffern, die sich berühren oder überlappen (siehe Abbildung 5.1).

Die Entwicklung eines Paarerkenners wurde von Dieter Niggeler in dessen Diplomarbeit vorgeschlagen [Nig94]. Als Bestandteil eines Zahlenerkenners sollte dieser dazu beitragen, die Problematik der Segmentierung von Zahlen zu entschärfen.

Anhand einer publizierten Messung[1] enthalten etwa 20% der handgeschriebenen Postleitzahlen Ziffern, die sich berühren oder überlappen und hier Ziffergruppen

[1] Die Messungen wurden anhand von 479 Zahlenbildern der CEDAR Datenbank durchgeführt[KS92].

Abbildung 5.1: Zwei Bilder von Zifferpaaren mit den Wahrheitswerten '45' und '00'

Zifferngruppe	Anzahl Gruppen	betroffene Zahlen
Ziffernpaar	92	(17.1%)
Drei Ziffern	13	(2.5%)
Vier Ziffern	2	(0.4%)
Fünf Ziffern	1	(0.2%)
Total	108	(20.3%)

Tabelle 5.1: Gemessene Häufigkeiten verschiedener Ziffergruppen

genannt werden sollen. Ziffergruppen können nach der Anzahl der darin enthaltenen Ziffern unterschieden werden, wobei aus zwei Ziffern bestehende Zifferngruppen auch Ziffernpaare genannt werden.

Die gemessenen Häufigkeiten verschiedener Zifferngruppen werden in der Tabelle 5.1 wiedergegeben. Mit Abstand am häufigsten (über 85% der vorkommenden Zifferngruppen) wurden Ziffernpaare beobachtet, womit auch die praktische Relevanz eines Systems zur Erkennung von Zifferpaaren gegeben ist.

Da (noch) keine sehr zuverlässigen Verfahren zur Segmentierung von Zifferngruppen existieren, besteht die Idee des Paarerkenners darin, auf eine explizite Segmentierung zu verzichten, und Ziffernpaare als Einheiten zu trainieren und zu erkennen.

5.2 Generierung von Paarbildern

Um einen Paarerkenner trainieren und testen zu können, müssen genügend Beispiele von Ziffernpaaren verfügbar sein. Im Gegensatz zur Ziffernerkennung kann bei der Paarerkennung nicht direkt auf bestehende Bilddaten zurückgegriffen werden, da keine spezifischen Datenbanken zur Paarerkennung existieren.

Aus folgenden Gründen scheint es nicht praktikabel, Ziffernpaare aus Zahlen

nachträglich zu extrahieren: Tausende von Zahlenbildern müssten einzeln darauf geprüft werden, ob darin Ziffernpaare vorhanden sind. Die so entdeckten Ziffernpaare müssten aus dem Kontext der Zahl extrahiert und mit dem entsprechenden Wahrheitswert versehen werden. Als Beispiel hierfür wurde das Ziffernpaar '45' in Abbildung 5.1 manuell aus der Zahl '16745' der Abbildung 3.4 extrahiert (und binärisiert).

Neben der limitierten Zeit zur Beschaffung von Ziffernpaaren stellt auch der erforderliche Umfang einer Ziffernpaar-Datenbank ein grosses Problem dar. Um für die Paarerkennung eine zur Ziffernerkennung vergleichbare Situation zu schaffen, müssten zum Training des Paarerkenners tausende von Beispielen pro Paarklasse vorhanden sein, was einem Umfang von mehreren 100'000 Zifferpaaren entsprechen würde. Aus der gesamten CEDAR Datenbank könnten jedoch nur etwas 2000 Ziffernpaare extrahiert werden, die zum Training und Testen eines Paarerkenners verwendet werden könnten.

Als einziger Ausweg aus der dargelegten Situation bot sich eine künstliche Erzeugung von Ziffernpaaren an[2], obwohl dies im Widerspruch zu den Anforderungen an Bilddaten steht. Wie in Abschnitt 3.1 beschrieben, sollten Bilddaten aus dem Umfeld stammen, das für den geplanten Einsatz des Erkennungssystems relevant ist. Künstlich generierte Ziffernpaare werden jedoch höchstens zufälligerweise repräsentativen Charakter für "natürliche" Ziffernpaare haben.

Zur künstlichen Erzeugung eines Ziffernpaares werden mit Vorteil zwei handgeschriebene Ziffern kombiniert, die von derselben Person stammen. Die NIST SD3 Datenbank schien für diesen Zweck die optimale Quelle, da diese von 2100 Personen für jeweils alle Ziffernklassen mehrere Beispiele enthält.

Um die bei "natürlichen" Ziffernpaaren beobachtete Variabilität der Zusammensetzung zu modellieren, wurden die drei Parameter G_{rel}, V_{abs} und H_{abs} zur künstlichen Generierung der Ziffernpaare eingeführt, durch welche die konkrete Kombination der Ziffern beeinflusst werden kann. Mit dem Parameter $G_{rel} \in [0.5, 2.0]$ kann das Verhältnis der Grösse der ersten Ziffer zur Grösse der zweiten Ziffer bestimmt werden, mit dem Parameter V_{abs} kann die relative, vertikale Position der Mitten der beiden Ziffern in Anzahl Bildpunkten eingestellt werden, und mit dem dritten Parameter H_{abs} wird die absolute, horizontale Überlappung der beiden Ziffern (ebenfalls in Anzahl Bildpunkten) angegeben.

In Abbildung 5.2 ist das Vorgehen zur Konstruktion eines Ziffernpaares mit dem Wahrheitswert '40' und den folgenden Parameterwerten dargestellt: $G_{rel} = 1.5$, $V_{abs} = 13$ und $H_{abs} = 3$.

Nach der Auswahl der beiden Ziffern wird die erste Ziffer auf die 1.5-fache Grösse der zweiten Ziffer skaliert, so dass die Höhe der ersten Ziffer 43 Bildpunkte und

[2]Die künstliche Erzeugung von Ziffernpaaren wurde auch in [KRL91] beschrieben.

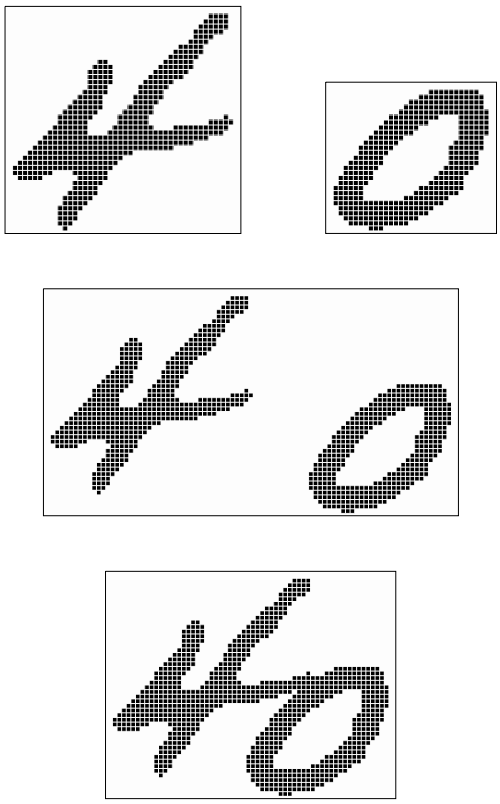


Abbildung 5.2: Konstruktion eines Ziffernpaares mit dem Wahrheitswert '40'

die Höhe der Zweiten Ziffer 28 Bildpunkte beträgt (1. Zeile der Abbildung 5.2), anschliessend werden die beiden Ziffern gemäss der gewünschten relativen Position so in ein leeres Bild kopiert, dass die Mitte der zweiten Ziffer 13 Bildpunkte tiefer als die Mitte der ersten Ziffer zu liegen kommt (2. Zeile) und zum Schluss werden die beiden Ziffern horizontal Bildpunkt um Bildpunkt zusammengeschoben, bis die gewünschte Überlappung von 3 Bildpunkten erreicht ist (3. Zeile).

5.3 Aufbau

Da der Paarerkenner auf der Architektur und den Verfahren des im letzten Kapitel vorgestellten Ziffernerkenners basiert, werden in diesem Abschnitt hauptsächlich Unterschiede zum Ziffernerkenner behandelt. Die Architektur des Paarerkenners ist in der Abbildung 5.3 dargestellt.

Die Unterschiede beschränken sich auf die folgenden drei Punkte: In der Vorverarbeitung wird zusätzlich eine Normierung der Schriftneigung durchgeführt, die Klas-

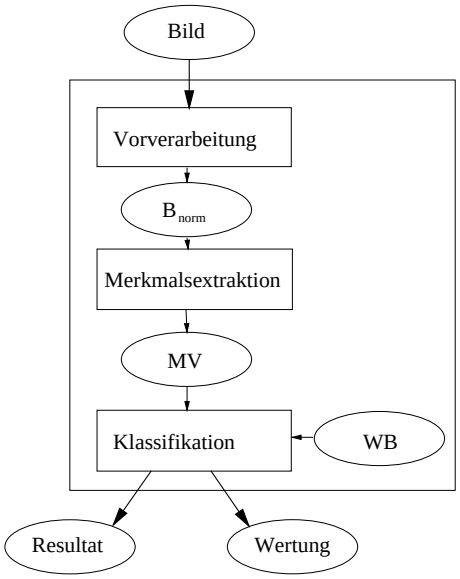


Abbildung 5.3: Aufbau des Paarerkenners

sifikation verwendet ein dem 'Nearest Neighbour' verwandtes Verfahren und für das Training des Paarerkenners, wurde ein Verfahren entwickelt, das während dem Trainingsvorgang bereits vorhandenes Wissen der Wissensbasis berücksichtigt.

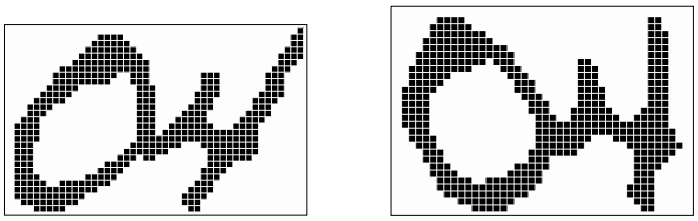
5.3.1 Normierung der Schriftneigung

Durch die Normierung der Schriftneigung kann der Handschrift einen Teil der Variabilität genommen werden, was ihre Erkennung erleichtert. Experimentelle Ergebnisse haben bestätigt, dass sich durch diese zusätzliche Normierung in der Vorverarbeitung die Erkennungsrate des Paarerkenners signifikant verbessern lässt.

Anhand eines generierten Ziffernpaares demonstriert Abbildung 5.4 den Effekt der Normierung der Schriftneigung. Da die Schriftneigung von Person zu Person variiert, muss diese für jedes Ziffernpaar geschätzt werden. Nachdem die Schätzung der Schriftneigung vorliegt, kann das Bild durch eine horizontale Scherung so transformiert werden, dass die Schriftneigung in die Vertikale übergeht.

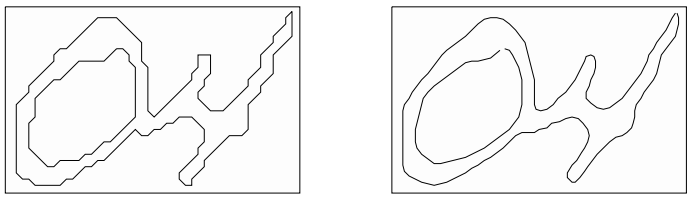
Das implementierte Verfahren zur Messung der Schriftneigung orientiert sich dabei an der Konturlinie eines Ziffernpaares[3]. Es kann in drei Schritte unterteilt werden. In einem ersten Schritt werden die Konturlinien des Ziffernpaares extrahiert und

[3]Ein ähnliches Verfahren wurde in [CGM93] publiziert und von Peter Steiner in seiner Diplomarbeit zur Normierung der Schriftneigung von handgeschriebenen Wörtern verwendet [Ste95].



(a) (b)

Abbildung 5.4: Beispiel eines Ziffernpaares vor (a) und nach (b) der Normierung der Schriftneigung



(a) (b)

Abbildung 5.5: Extrahierte Konturen (a) und geglättete Konturen (b) eines Ziffernpaares

geglättet. Abbildung 5.5 zeigt die aus Abbildung 5.4(a) extrahierten und geglätteten Konturen.

Im zweiten Schritt wird ein Richtungshistogramm für die geglättete Kontur berechnet und dessen Median bestimmt. Es hat sich gezeigt, dass der Median des Richtungshistogrammes eine stabile Schätzung der Schriftneigung darstellt.

Abbildung 5.6 stellt das Richtungshistogramm für die geglättete Kontur der Ab-

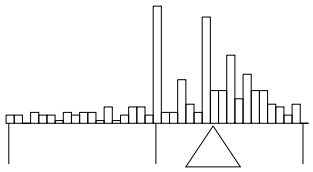


Abbildung 5.6: Richtungshistogramm der geglätteten Konturen

i	WW	Abstand	Wertung
1	'04'	5	0.1667
2	'84'	8	0.1111
3	'04'	9	0.1000
4	'06'	18	0.0526
...			
10	'62'	43	0.0227

Tabelle 5.2: Beispiel einer Zwischentabelle zur Klassifikation von Ziffernpaaren

bildung 5.5(b) dar[4]. Der Median des Histogramms ist in der Abbildung durch ein Dreieck markiert. Somit wird für das Ziffernpaar (a) in Abbildung 5.4 eine Schriftneigung von 30 Grad (nach rechts) geschätzt.

Im letzten Schritt der Normierung der Schriftneigung muss das Ziffernpaar nur noch um den negativen, geschätzten Neigungswinkel durch eine horizontale Scherung transformiert werden. Das Resultat dieser Scherung wird in Abbildung 5.4(b) dargestellt.

5.3.2 Klassifikation

Das eingesetzte Klassifikationsverfahren des Paarerkenners basiert auf der 'Distance Weighted k-Nearest Neighbour' – oder kurz DWkNN-Methode[5] und lässt sich in zwei Schritte unterteilen.

Um einen Merkmalsvektor zu klassifizieren, werden in einem ersten Schritt die zehn nächsten Prototypen der Wissensbasis des Paarerkenners[6] bestimmt und deren Wahrheitswerte zusammen mit den berechneten Absänden in eine Zwischentabelle eingetragen. Jedem Eintrag der Zwischentabelle wird anschliessend die Wertung $1/(1 + Abstand)$ zugeordnet.

Tabelle 5.2 gibt ein Beispiel für eine Zwischentabelle, die bei der Klassifikation des Ziffernpaares (a) in Abbildung 5.4 enstanden sein könnte.

Im zweiten Schritt des Klassifikationsverfahrens wird anhand der Zwischentabelle

[4]Das linke Ende des Histogrammes entspricht der Konturrichtung von 90 Grad nach links und das rechte Ende der Konturneigung von 90 Grad nach rechts (bezüglich der Senkrechten).

[5]Dieses Verfahren wurde von S.A. Dudani 1976 publiziert [Dud76].

[6]Die Merkmalsextraktion ist bis auf die Aufteilung der Bilder in Regionen identisch zum im Kapitel 4 beschriebenen Ziffernerkenner. Deshalb weist die Wissensbasis des Paarerkenners dieselbe Struktur wie beim Ziffernerkenner auf.

i	WW	Wertung
1	'0'	0.1667
2	'8'	0.1111
3	'0'	0.1000
4	'0'	0.0526
...		
10	'6'	0.0227

Tabelle 5.3: Auszug aus der Zwischentabelle zur Bestimmung des Erkennungsresultates der ersten Ziffer

ein Erkennungsresultat für die erste Ziffer und ein Erkennungsresultat für die zweite Ziffer bestimmt. Diese beiden Resultate werden anschliessend zum Resultat des Paarerkenners kombiniert.

Die Bestimmung des Erkennungsresultates für die erste Ziffer wird anhand des Beispiels der Tabelle 5.2 erläutert, wobei Tabelle 5.3 diejenigen Informationen enthält, die zur Bestimmung des Erkennungsresultates der ersten Ziffer notwendig sind.

Tabelle 5.3 wird nun nach dem DWkNN-Verfahren ausgewertet. Dazu werden die Wertungen für jede vorkommende Ziffernklasse aufsummiert. Diejenige Klasse, welche die höchste Summe erreicht, gilt als Erkennungsresultat. Für das Beispiel in Tabelle 5.3 erreicht die Ziffernklasse '0' mit $0.1667 + 0.1000 + 0.0526 = 0.3193$ die höchste Summe und gilt somit als Erkennungsresultat für die erste Ziffer des Paares. Für die zweite Ziffer erreicht die Ziffernklasse '4' mit 0.3778 die höchste Summe womit das Resultat des Paarerkenners durch die Paarklasse '04' und die zugehörigen (gemittelten) Wertung von 0.3486 gebildet werden kann.

5.3.3 Inkrementelles Training

Mit 100 Klassen ist beim Paarerkenner die Anzahl der möglichen Interpretationen eines Bildes zehn mal grösser als beim Ziffernerkenner. Damit in der Wissensbasis des Paarerkenners jede Klasse durch eine umfangreiche Sammlung von Merkmalsvektoren repräsentiert wird, ist ein entsprechend aufwendiges Training erforderlich.

Da bei NN-basierenden Klassifikationsverfahren die Wissensbasis linear zur Anzahl trainierter Beispiele anwächst, und die benötigte Zeit zur Klassifikation eines Merkmalsvektors proportional zu deren Grösse ist, stösst man mit solchen Verfahren schnell an die Leistungsgrenzen der verfügbaren Computer.

Ein kurzes Rechenbeispiel soll diese Problematik veranschaulichen. Mit der NIST

SD3 Datenbank werden Ziffern von 2100 Personen zur Verfügung gestellt, die zu Ziffernpaaren kombiniert werden können. Um eine angemessene Variabilität[7] der generierten Ziffernpaare zu erreichen, sollten Ziffern von 500 Personen berücksichtigt werden, und pro Paarklasse und Person jeweils 10 unterschiedliche Beispiele generiert werden.

Um diesen Anforderungen gerecht zu werden, müsste der Paarerkenner mit einer halben Million ($100 \times 500 \times 10$) Ziffernpaaren trainiert werden[8]. Damit der benötigte Speicherplatz für die Wissensbasis und die zur Erkennung benötigte Rechenzeit in Grenzen gehalten werden konnten, wurde der Paarerkenner inkrementell trainiert[9].

Im Gegensatz zum normalen Training eines NN-basierenden Klassifikators, der alle aus den Trainingsbildern extrahierten Merkmalsvektoren (mit den zugehörigen Wahrheitswerten) als Prototypen in der Wissensbasis speichert, wird beim inkrementellen Training ein Merkmalsvektor nur dann als Prototyp in die Wissensbasis aufgenommen, wenn er aufgrund der bestehenden Wissensbasis mit der Nearest-Neighbor-Methode nicht korrekt klassifiziert werden kann.

Der Zuwachs der Wissensbasis steigt somit beim inkrementellen Training nicht mehr linear mit der Anzahl der verwendeten Trainingsbilder an, sondern nimmt mit zunehmender Grösse der aufgebauten Wissensbasis ab. Im folgenden Abschnitt wird auf diesen Punkt anhand eines konkreten Experimentes eingegangen.

5.4 Experimente und Resulate

In diesem Abschnitt werden fünf mit dem Paarerkenner durchgeführte Experimente zusammen mit den daraus gewonnenen Resultaten dokumentiert. Die ausgewählten Experimente sind dabei chronologisch nach deren Durchführung geordnet und behandeln das inkrementelle Training, die Optimierung der Merkmalsextraktion, die Klassifikation, die Normierung der Schriftneigung und das Lernverhalten des Paarerkenners.

Die folgenden Bemerkungen gelten für alle hier dokumentierten Experimente und werden deshalb in deren Beschreibung bis auf Abweichungen nicht mehr explizit erwähnt. Alle dokumentierten Erkennungsraten beziehen sich auf die Erkennung von Ziffernpaaren. Beispiele, bei denen nur eine der beiden Ziffern richtig erkannt wird, gelten nicht als richtig erkannt.

[7] Die angemessene Variabilität bezieht sich auf die Wahrscheinlichkeit, Ziffernpaare zu erzeugen, die mit "natürlichen" Ziffernpaaren vergleichbar sind.

[8] Auch das in [KRL91] vorgestellte Erkennungssystem wurde mit mehreren 100000 Beispielen von Ziffernpaaren trainiert.

[9] Andere Verfahren, welche Reduktion der Grösse der Wissensbasis zum Ziel haben, sind in [Das90] unter dem Kapitel 'Editing Experiments' angegeben.

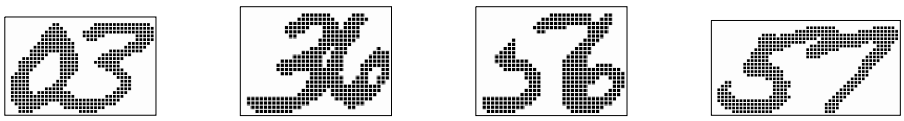


Abbildung 5.7: Beispiele für künstlich erzeugte Ziffernpaare

Wie in Abschnitt 5.2 besprochen, ist ein umfangreiches Training des Paarerkenners mit künstlichen Zifferpaaren erforderlich, wobei möglichst viele verschiedene Kombinationen von Ziffern berücksichtigt werden sollten. In der Praxis hat dies erhebliche Probleme wegen des erforderlichen Speicherbedarfes und der Rechenzeiten verursacht, denen mit starken Restriktionen bei der Generierung der Ziffernpaare begegnet werden musste.

Für jedes Formular der NIST SD3 Datenbank wurde nur ein einziges Ziffernpaar pro Paarklasse generiert (100 Ziffernpaare pro Formular). Um ein Ziffernpaar einer bestimmten Paarklasse zu generieren, wurden zwei Ziffern zufällig aus den verfügbaren Ziffern des Formulares ausgewählt, auf die gleiche Grösse skaliert und ohne vertikale Versetzung zusammengesetzt. In Abbildung 5.7 sind einige Beispiele so konstruierter Ziffernpaare enthalten, die zum Training und für die Tests des Paarerkenners verwendet wurden.

Die Aufteilung der 2100 Formulare in Trainings- und Testdaten entspricht den Angaben in Abschnitt 3.3 für die NIST SD3 Datenbank, wobei zum Training 49950 Paare aus Ziffern der Formulare f0000 bis f1499 und für die Tests 8136 Paare aus Ziffern der Formulare f2016 bis f2099 verwendet wurden.

Die dokumentierten Resultate beziehen sich auf die folgende Konfiguration des Paarerkenners: Normierung der Schriftneigung in der Vorverarbeitung, Aufteilung des Bildes in 3 (vertikal) mal 4 (horizontal) Regionen in der Merkmalsextraktion, Einsatz des neuen Klassifikators und des inkrementellen Trainings.

5.4.1 Inkrementelles Lernen

Das erste Experiment demonstriert den Einsatz des inkrementellen Trainings anhand eines Vergleiches. Dazu wurde der Paarerkenner zwei mal mit den identischen 50000 Ziffernpaaren[10] trainiert. Im ersten Trainingslauf wurde das klassische Trainingsverfahren angewendet und somit alle 50000 aus den Trainingsbeispielen extrahierten Prototypen (Prot.) in der Wissensbasis aufgenommen. Im zweiten Trainingslauf kam das inkrementelle Trainingsverfahren zur Anwendung. Vor dem eigentlichen inkre-

[10] Ziffern der Formulare f0000 bis f0499.

Trainings-stand	konventionelles Training	inkrementelles Training
bis f0499	49950 Prot. 85.01%	17065 Prot. 84.75%

Tabelle 5.4: Grössen der Wissensbasen und Erkennungsraten für konventionelles und inkrementelles Training

mentellen Training wurde die Wissensbasis mit den ersten 100 extrahierten MV's[11] initialisiert, anschliessend wurde jeder zusätzlich extrahierten MV mit der bisherigen Wissensbasis klassifiziert. Nur bei einer Fehlklassifikation wurde der entsprechende MV zusammen mit seinem Wahrheitswert als Prototyp in die Wissensbasis aufgenommen. Nach dem Abschluss des inkrementellen Trainings wies die Wissensbasis nur 17000 MV auf. Die Ergebnisse dieses Experiments sind in Tabelle 5.4 zusammengefasst.

Getestet wurden die beiden unterschiedlich trainierten Paarerkenner anhand von 8136 Paaren, die aus Ziffern der Formulare f2016 bis f2099 generiert worden sind. Der inkrementell trainierte Paarerkenner erreichte eine Erkennungsrate von 84.75%. Der klassisch trainierten Paarerkenner erreichte mit einer drei Mal grösseren Wissenbasis eine um nur 0.26% höhere Erkennungsrate.

5.4.2 Merkmalsextraktion

Die für den Ziffernerkenner verwendete Aufteilung der Konturbilder in 3×3 Regionen in der Merkmalsextraktion hat sich für den Paarerkenner als nicht optimal herausgestellt. Um die optimale Unterteilung der Konturbilder zu bestimmen, wurden Messungen für die Aufteilung in 3v/2h, 3v/3h, 3v/4h und 3v/5h Regionen durchgeführt, wobei mit '3v/2h' die Aufteilung in 3 (vertikal) mal 2 (horizontal) Regionen bezeichnet wird. In Tabelle 5.5 sind die Erkennungsraten (ohne Normierung der Schriftneigung) sowie die Grössen der Wissensbasen für den Trainingsstand nach den ersten 100, 200 und 500 Formularen wiedergegeben.

Aufgrund der in Tabelle 5.5 präsentierten Ergebnissen wurde eine Aufteilung der Konturbilder in 3 (vertikal) mal 4 (horizontal) Regionen gewählt. Die Aufteilung in 3 mal 2 Regionen führte bereits zu Beginn zu deutlich schwächeren Resultaten und wurde deshalb nicht weiter erfolgt.

[11] Ziffern des Formulares f0000.

Trainings-stand	3v/2h	3v/3h	3v/4h	3v/5h
bis f0099	8683 Prot. 16.20%	5799 Prot. 68.30%	5761 Prot. 70.80%	6046 Prot. 63.73%
bis f0299		10935 Prot. 74.30%	10859 Prot. 73.80%	11372 Prot. 70.10%
bis f0499		24044 Prot. 78.50%	24219 Prot. 80.01%	25517 Prot. 73.90%

Tabelle 5.5: Grössen der Wissensbasen und Erkennungsraten für unterschiedliche Aufteilung des Konturbildes in der Merkmalsextraktion

Trainings-stand	NN	DWkNN	neu
bis f0499	71.88%	83.95%	84.76%
bis f0999	73.69%	85.15%	85.83%
bis f1499	76.27%	86.75%	87.29%

Tabelle 5.6: Erkennungsraten für unterschiedliche Klassifikatoren

5.4.3 Klassifikation

Im nachfolgenden Experiment werden die Erkennungsraten für verschiedene Klassifikatoren miteinander verglichen. Als Klassifkatoren wurden der NN-Klassifikator, der DWkNN-Klassifikator und der neue, in Abschnitt 5.3 beschriebene Klassifikator verwendet. Die gemessenen Erkennungsraten sind in Tabelle 5.6 für den Trainingsstand nach den ersten 500, 1000 und 1500 Formularen wiedergegeben.

Die Resultate zeigen, dass die besten Erkennungsraten immer vom neu entwickelten Klassifikator erreicht wurden, wenn auch dessen Vorsprung mit zunehmendem Training abnahm.

5.4.4 Normierung der Schriftneigung

Die Einführung der Normierung der Schriftneigung in der Vorverarbeitung des Paarerkenners erzielte erhebliche Verbesserungen, die im nächsten Experiment dokumentiert sind.

Die Normierung der Schriftneigung der Ziffernpaare hatte eine Verkleinerung der Variabilität innerhalb der Paarklassen zur Folge. Die in Tabelle 5.7 zusammengefassten

Trainings-stand	ohne Normierung	mit Normierung
bis f0499	24219 Prot. 80.01%	17065 Prot. 84.75%
bis f0999	43952 Prot. 80.60%	30845 Prot. 85.83%

Tabelle 5.7: Grössen der Wissensbasen und Erkennungsraten des Paarerkenners mit und ohne Normierung der Schriftneigung

Resultate belegen, dass diese Variabilität für den notwendigen, hohen Trainingsaufwand zumindest mitverantwortlich ist.

Durch diese Normierung konnte eine Verbesserung der Erkennungsrate von über 5% bei einer gleichzeitigen Reduktion der Grösse der Wissensbasis um 30% erreicht werden.

5.4.5 Lernverhalten

Das letzte, hier beschriebene Experiment sollte das Lernverhalten des Paarerkenners während des inkrementellen Trainings genauer untersuchen. Dazu wurde nach jeweils 100 trainierten Formularen gemessen, wieviele der jeweils 100 (pro Formular) extrahierten Merkmalsvektoren als Prototypen zur Wissensbasis hizugefügt werden mussten. Da die gemessenen Zuwachsrachsraten sehr stark von den einzelnen Formularen abhängig sind, wurden die Werte von jeweils fünf Formularen gemittelt und deren empirische Standardabweichung bestimmt.

Abbildung 5.8 zeigt die so erhaltene Lernkurve, wobei jeweils die berechneten Mittelwerte miteinander verbunden wurden und die Standardabweichungen nach oben und unten als Fehlerbalken eingetragen wurden.

Tabelle 5.8 fasst die Grösse der Wissensbasis für den Trainingsstand nach 200, 500, 1000 und 1500 Formularen sowie die zugehörigen Erkennungsraten des Paarerkenners für dieses Experiment zusammen. Die gemessenen Werte für die Grösse der Wissensbasis bestätigen, dass diese beim inkrementellen Training nicht mehr linear mit der Anzahl trainierter Beispiele anwächst.

Das gemessene Lernverhalten zeigt für die ersten drei Messpunkte der Abbildung 5.8 klar die erwartete, kontinuierliche Abnahme der Zuwachsrate bei zunehmender Grösse der Wissensbasis. Die folgenden Messpunkte schwanken jedoch sehr stark um den (abgeschätzten) Wert von 30 Prototypen bei 700 trainierten Formularen und 20 Prototypen bei 1500 trainierten Formularen.

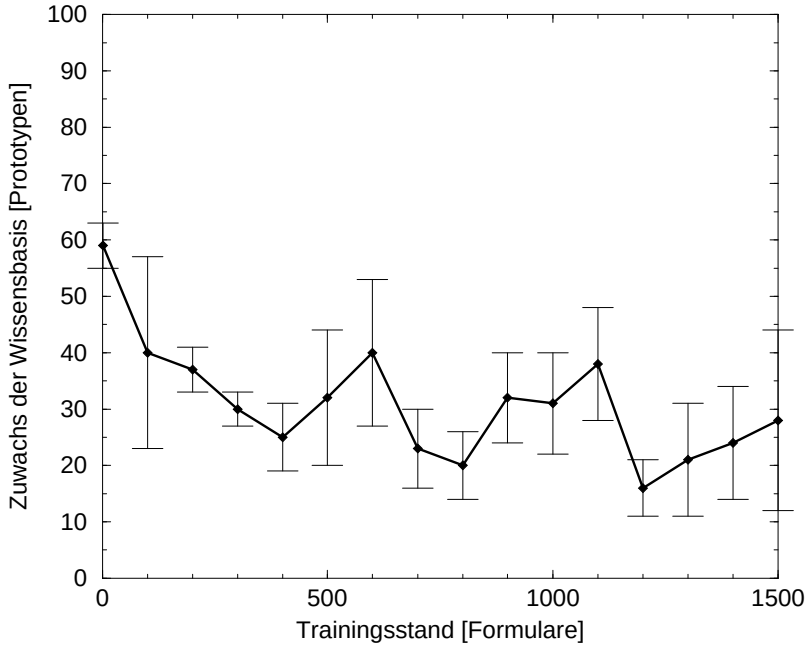


Abbildung 5.8: Verlauf der Zuwachsrate der Wissensbasis während dem Training

Um einen weiteren Abfall der Zuwachsrate auf etwa 10 Prototypen pro trainiertem Formular zu erreichen, hätten Ziffernpaare von schätzungsweise 1000 bis 2000 weiteren Formularen trainiert werden müssen, was aus Gründen der verfügbaren Speicher- und Rechenkapazität jedoch praktisch nicht mehr möglich war.

Trainings-stand	Grösse der Wissensbasis	Erkennungsrate
bis f0199	7776 Prot.	82.15%
bis f0499	17065 Prot.	84.76%
bis f0999	30845 Prot.	85.83%
bis f1499	43190 Prot.	87.29%

Tabelle 5.8: Entwicklung der Grösse der Wissensbasis und Erkennungsrate während des Trainings

5.5 Diskussion

Die im letzten Abschnitt präsentierten Expererimente belegen, dass sich ein bewährter Ziffernerkenner nicht durch einige kleine Anpassungen in einen zuverlässigen Paarerkenner umbauen lässt. Trotz der Verwendung von künstlichen, stereotyp konstruierten Zifferpaaren und der Durchführung eines umfangreichen Trainings (150000 Beispiele), lag die Erkennungsrate des vorgestellten Paarerkenners bei nur 87.3%.

Die erreichte Erkennungsrate ist erst durch die Einführung des inkrementellen Trainings und der Normierung der Schriftneigung möglich geworden. Die Normierung der Schriftneigung hat eine Verbesserung der Erkennungsrate um 5% zur Folge gehabt. Dank des inkrementellen Trainings konnte die Grösse der Wissensbasis so stark reduziert werden, dass die vorgestellten Experimente mit den vorhandenen Computern überhaupt durchführbar waren.

Ein Vergleich der erzielten Resultate mit anderen Arbeiten ist sehr schwierig, da über die Erkennung isolierter Ziffernpaaren praktisch nichts publiziert worden ist. Mir ist lediglich eine Arbeit von James D. Keeler, David E. Rumelhart und Wee-Kheng Leow [KRL91] bekannt, die sich ebenfalls mit der Erkennung künstlich generierter Ziffernpaare[12] beschäftigt. Im Gegensatz zu den in dieser Diplomarbeit generierten Ziffernpaaren berührten oder überlappten sich die beiden Ziffern bei [KRL91] in nur etwa 15% der Fälle. Als Erkennungssystem wurde ein sogenanntes 'Time Delayed Neuronal Network' eingesetzt[13], das mit einigen 100000 generierten Paaren trainiert wurde. Die von diesem System erreichte Erkennungsrate lag bei 81%.

5.6 Schlussfolgerungen

Aufgrund von Messungen der CEDAR Datenbank enthalten etwa 20% der handgeschriebenen Zahlen Ziffern, die sich berühren, wobei es sich in 80% dieser Fälle um Ziffernpaare handelt. Da es bisher keine wirklich zuverlässigen Verfahren gibt, um Ziffernpaare in ihre Ziffern zu segmentieren, führen Ziffernpaare immer wieder zu Erkennungsfehlern.

Für Ziffernpaare sollte das Problem der Segmentierung dadurch gelöst werden, dass ein Paarerkenner auf eine explizite Segmentierung verzichtet und Ziffernpaare als Einheiten erkennen kann.

Die beiden Hauptprobleme, die sich im Laufe dieser Diplomarbeit herauskristallisiert

[12] Die Herkunft der verwendeten Ziffern wurde in der Arbeit nicht genau spezifiziert, sie stammen jedoch nicht aus der NIST SD3 Datenbank.

[13] Urs-Viktor Marti hat dieses System in seiner Diplomarbeit eingehend beschrieben [Mar96].

haben, können folgendermassen zusammengefasst werden:

- Um einen Paarerkenner trainieren und testen zu können, sind Beispiele von Ziffernpaaren erforderlich. Die Bereitstellung "natürlicher" Ziffernpaare ist jedoch aus den folgenden Gründen nicht realistisch: Erstens stehen potentiell nicht genügend Ziffernpaare zur Verfügung und zweitens wäre deren Extraktion aus den vorhandenen Zahlen zu aufwendig. Aus dieser Überlegung folgt, dass die Ziffernpaare generiert werden müssen und somit ein so trainierter Paarerkenner seine optimale Leistungsfähigkeit auch nur für künstliche Ziffernpaare erreichen kann.
- Der erforderliche Aufwand um einen Paarerkenner zu trainieren ist extrem hoch. Im Vergleich zur Ziffernerkennung ist bei der Paarerkennung die Variabilität innerhalb der Klassen als auch die Anzahl der zu unterscheidenden Klassen viel höher.

Im Gegensatz zum ersten Hauptproblem, das eine Randbedingung darstellte und somit in dieser Diplomarbeit nicht gelöst werden konnte, sind vor allem durch das inkrementelle Training des Paarerkenners und durch die Einführung der Schriftneigungsnormierung Ansätze zur Lösung des zweiten Hauptproblems gefunden worden.

Um die Problematik der Paarerkennung weiter zu bearbeiten, sollte jedoch zuerst eine Datenbank mit "natürlichen" Ziffernpaaren aufgebaut werden. Eine solche Datenbank könnte zumindest beim Testen von Paarerkennern eingesetzt werden. Somit würden realistische Schätzungen von Erkennungsraten möglich und echte Verbesserungen könnten messbar gemacht werden.

Kapitel 6

Ein Zahlenerkenner

In diesem Kapitel wird das in dieser Diplomarbeit implementierte System zur off-line Erkennung handgeschriebener Zahlen vorgestellt. Die Ausführungen dieses Kapitels beschäftigen sich hauptsächlich mit der Architektur dieses Systems und der Segmentierung von Zahlen. Das hier vorgestellte System setzt die Verfügbarkeit eines Ziffernerkenner voraus, der selbst nicht Gegenstand dieser Diplomarbeit war.

Im ersten Abschnitt wird die zu lösende Aufgabe vorgestellt. Im Abschnitt 6.2 werden zwei Ansätze zur Segmentierung von handgeschriebenen Zahlen behandelt. Die Systenmarchitektur des Zahlenerkenners wird im Abschnitt 6.3 im Überblick erläutert, und die einzelnen Module des implementierten Zahlenerkenners werden in den Abschnitten 6.4 bis 6.7 genauer beschrieben.

6.1 Aufgabenstellung

Der Zahlenerkenner soll einem gegebenen Bild einer Zahl beliebiger Länge die korrekte Interpretation zuorden können. Der Zahlenerkenner soll fähig sein, die Korrektheit der berechneten Interpretation abzuschätzen und, falls diese Abschätzung ein vorgegebenes Mass unterschreitet, das zugehörige Bild zurückweisen.

Diese Aufgabenstellung weicht in den folgenden beiden Punkten von der Aufgabenstellung des Ziffernerkenners und des Paarerkenners ab.

- Es existiert keine fixe Anzahl von möglichen Interpretationen.
- Um die Wahrscheinlichkeit von Fehlinterpretationen zu minimieren, sollen Eingaben zurückgewiesen werden können.

Abbildung 6.1: Schema des Zahlerkenners

Abbildung 6.1 stellt ein Schema des Zahlerkenners gemäss der Aufgabenstellung dar, wobei sich die beiden Ausgänge 'Resultat' und 'Rückweisung' gegenseitig ausschliessen.

6.2 Segmentierung von Zahlen

Wie bereits im Kapitel 2 angesprochen, stellt die korrekte Segmentierung von Wörtern oder Zahlen ein zentrales Problem der Handschrifterkennung dar. Da bei der Erkennung von Zahlen fast jedes Resultat (zumindest potentiell) möglich ist, muss jede Ziffer der Zahl mit genügender Sicherheit korrekt erkannt werden können, was an die Segmentierung von Zahlen sehr hohe Ansprüche stellt[1].

Um die verschiedenen Probleme der Segmentierung von Zahlen in ihre Ziffern anschaulich zu erläutern, soll hier der Begriff der Zusammenhangskomponente (ZK) eingeführt werden. Zusammenhangskomponenten bestehen aus zusammenhängenden Mengen von schwarzen Bildpunkten auf einem Hintergrund von weissen Bildpunkten.

(a) (b) (c)

Abbildung 6.2: Zusammenhangskomponenten und Segmentierung

Wie in der Abbildung 6.2 durch Beispiele dargestellt, muss jede Methode zur Segmentierung von Zahlen mit den folgenden Situationen umgehen können.

[1] Wenn bei der Erkennung von Wörtern einige Buchstaben nicht richtig erkannt werden, ist eine korrekte Erkennung des Wortes anhand des Kontextes oder eines Wörterbuches meistens trotzdem noch möglich.

- Fall (a): Eine ZK stellt genau eine Ziffer dar. Durch eine Extraktion der ZK's wird die korrekte Segmentierung der Zahl '64' erreicht.
- Fall (b): Eine ZK stellt einen Teil einer Ziffer dar (die Ziffer '5' besteht aus 3 ZK's). Durch die Extraktion der ZK's erfolgt eine Übersegmentierung der Ziffer '5'.
- Fall (c): Eine ZK stellt eine Zifferngruppe dar (die drei Ziffern '4', '0' und '9' berühren sich und verschmelzen so zu einer ZK). Durch die Extraktion der ZK's würde diese Zifferngruppe untersegmentiert.

Neben den oben aufgeführten Situationen kann ein Bild auch aus einer beliebigen Kombination dieser Fälle bestehen. Zwei grundsätzlich verschiedene Ansätze zur Segmentierung von Zahlen, die 'diskreten' Methoden und die 'kontinuierlichen' Methoden, werden in den beiden folgenden Abschnitten besprochen.

6.2.1 Der diskrete Ansatz

Die diskreten Methoden lassen sich in 'segmentierungsbasierte' und 'segmentierungsfreie' Verfahren unterteilen. Die segmentierungsbasierten Methoden zeichnen sich dadurch aus, dass sie ein Bild in einem ersten Schritt segmentieren und die resultierenden Teilbilder anschliessend als Ziffern interpretieren. Da bei den segmentierungsbasierten Methoden keine nachträgliche Verifikation der gefundenen Segmentierungsstellen erfolgt, werden solche Methoden auch als "blinde" Segmentierer bezeichnet. Eine solche Segmentierungsmethode stellt beispielsweise die 'Hit-and-Deflect' Strategie dar, welche Segmentierungsstellen anhand von oberen und unteren Konturpunkten des Zahlenbildes definiert [SNG92].

Der Nachteil segmentierungsbasierter Methoden besteht darin, dass sich Segmentierungsfehler direkt in Erkennungsfehlern auswirken. Die Vorteile liegen in der Einfachheit der Idee sowie der zur Anzahl Ziffern linearen Zeitkomplexität.

Sogenannt 'segmentierungsfreie' Methoden beruhen nicht auf der starren Reihenfolge erst Segmentierung, dann Erkennung, sondern bilden in einer ersten Phase lokale Hypothesen für Segmentierungsstellen, die anschliessend bewertet werden. Aus den vielen potentiellen Segmentierungsstellen wird in einer zweiten Phase eine optimale, global konsistente Teilmenge bestimmt. Ein segmentierungsfreies Verfahren, das auf der Detektion topologischer Merkmale wie Enden, Verzweigungen und Kreuzungen beruht ist in [NM92] vorgestellt[2].

[2]Dieses Verfahren diente als Grundlage zum Entwurf des Moduls 'Erkennung Teilbild' (siehe Abschnitt 6.6).

Der Vorteil segmentierungsfreier gegenüber segmentierungsbasierter Methoden besteht in derer Fähigkeit, auch Fälle korrekt zu segmentieren, die ein Wissen des Erkennungsresultates voraussetzen. Der Nachteil besteht darin, dass durch das Betrachten aller potentiellen Segmentierungsstellen die Zahl der möglichen Lösungen mit der Anzahl der Ziffern sehr stark ansteigt und somit die Wahl der (in der Regel einzigen) korrekten Segmentierung immer unwahrscheinlicher wird. Für Zahlen, die aus vielen Ziffern bestehen, bricht deshalb bei segmentierungsfreien Methoden die Erkennungsrate schnell zusammen.

6.2.2 Der kontinuierliche Ansatz

Im Gegensatz zum diskreten Ansatz, der nur einzelne (potentielle) Segmentierungsstellen berücksichtigt, wird bei kontinuierlichen Verfahren jede Bildspalte als potentielle Mitte einer Ziffer betrachtet. In diesen Methoden wird eine Maske mit einer vertikalen, streifenförmigen Aussparung von links nach rechts über das Bild geschoben und für jede Position der Maske der zugehörige Bildauschnitt bewertet. Beim Überqueren der Mitte einer Ziffer ergibt sich ein lokales Maximum der Bewertung. Somit können die Täler dieser Bewertungsfunktion als Segmentierungsstellen betrachtet werden. Eine solche Methode wird in [LL95] zur Segmentierung und Erkennung von Zahlen vorgestellt.

6.3 Systemarchitektur

Die Systemarchitektur des Zahlenerkenners ist in der Abbildung 6.3 schematisch dargestellt. Sie kombiniert eine segmentierungsbasierte und eine segmentierungsfreie Methode, um die Nachteile der einen Methode durch die Vorzüge der anderen zu kompensieren.

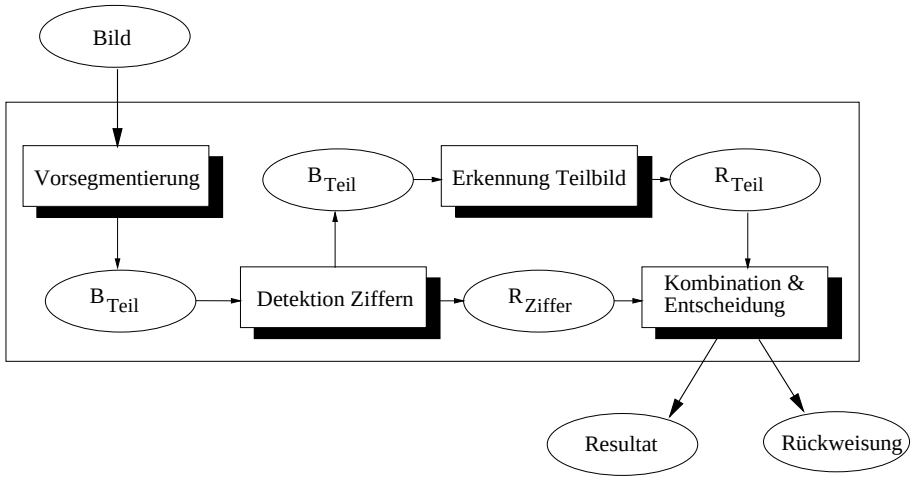


Abbildung 6.3: Aufbau des Zahlerkenners

Die Module 'Vorsegmentierung' und 'Detektion Ziffern' bilden den segmentierungsbasierten Teil der Erkennung. Für Spezialfälle wird der segmentierungsfreie Zahlerkenner 'Erkennung Teilbild' eingesetzt.

Im Modul 'Vorsegmentierung' wird das zu erkennende Bild in eine Sequenz von Teilbildern B_{Teil} zerlegt, die isolierte Ziffern oder Zifferngruppen enthalten. Das Modul 'Detektion Ziffern' überprüft alle Teilbilder darauf, ob sie isolierte Ziffern enthalten. Isolierte Ziffern werden direkt in diesem Modul erkannt und das zugehörige Erkennungsresultat R_{Ziffer} ans Modul 'Kombination & Entscheidung' gesandt. Falls ein Teilbild eine Zifferngruppe mit zwei oder mehr Ziffern enthält, wird dieses unverändert dem Modul 'Erkennung Teilbild' übergegeben, welches Zifferngruppen auch dann erkennen kann, wenn sich zwei oder mehr Ziffern berühren oder einzelne Ziffern aus mehreren Teilstücken bestehen. Das Resultat R_{Teil} der Teilbilderkennung wird anschliessend ebenfalls an das Modul 'Kombination & Entscheidung' übergeben. In diesem Modul werden die Resultate der Module 'Detektion Ziffern' und 'Erkennung Teilbild' zusammengesetzt und anhand einer Bewertung des Resultates entschieden, ob das zu erkennende Bild akzeptiert werden kann. Falls die Bewertung einen vorgegebenen Schwellwert überschreitet, wird das Resultat ausgegeben, andernfalls kommt es zu einer Rückweisung.

Die Grundidee dieser Architektur basiert auf der Beobachtung, dass die zuverlässige Segmentierung einer Zahl in Zifferngruppen viel einfacher zu realisieren ist, als eine direkte Segmentierung in isolierte Ziffern. Zur Segmentierung einer Zahl in Zifferngruppen reicht es aus, Bruchstücke von Ziffern oder Zifferngruppen zu detektieren (siehe Abbildung 6.2(b)) und diese mit ihrer Umgebung zu verknüpfen. Die aus dieser Segmentierung resultierenden Teilbilder enthalten dann entweder bereits isolierte Ziffern oder Zifferngruppen, die nur aus wenigen Ziffern bestehen. Isolierte Ziffern können direkt durch einen Ziffernerkenner mit sehr hoher Sicherheit erkannt werden. Zifferngruppen aus wenigen Ziffern können mit einer akzeptablen Erkennungsrate von einer segmentierungsfreien Methode erkannt werden.

Alle "einfachen" Bilder lassen sich mit diesem Ansatz korrekt segmentieren und mit einer hohen Zuverlässigkeit erkennen. Nur "komplexe" Teilstücke (Zifferngruppen) müssen durch aufwendigere Verfahren bearbeitet werden.

6.4 Modul Vorsegmentierung

Das Ziel der Vorsegmentierung besteht darin, das Eingabebild so in disjunkte Teilbilder B_{Teil} zu zerlegen, dass jedes Teilbild eine isolierte, vollständige Ziffer oder eine komplette Zifferngruppe enthält.

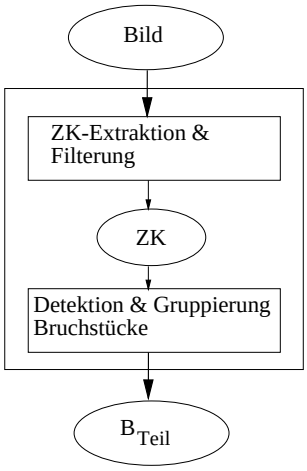
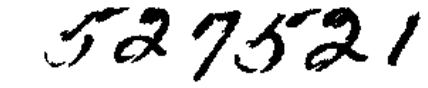


Abbildung 6.4: Aufbau des Modules Vorsegmentierung

Wie dies das Schema in Abbildung 6.4 veranschaulicht, läuft die Vorsegmentierung in zwei Phasen ab. In der ersten Phase werden alle ZK's (Zusammenhangskomponenten) des Bildes extrahiert und sehr kleine, nur aus wenigen Bildpunkten bestehende ZK's herausgefiltert.

In der zweiten Phase der Vorsegmentierung werden alle aus der ersten Phase resultierenden ZK's darauf untersucht, ob sie potentielle Bruchstücke von Ziffern darstellen. Als Bruchstück klassifizierte ZK's werden mit ihrer linken und/oder rechten Umgebung zu Teilbildern verknüpft. Bevor auf die konkrete Bestimmung von Bruchstücken und der Gruppierung von ZK's zu Teilbildern eingegangen wird, soll das erwartete Resultat der Vorsegmentierung durch Abbildung 6.5 veranschaulicht werden.
Die Klassifizierung einer ZK als potentielles Bruchstück ist durch die beiden folgenden Beobachtungen motiviert:

- Bis auf wenige Ausnahmen sind die Ziffern einer Zahl etwa so hoch, wie die Zahl selbst.
- Die Ziffern '4' und '5' werden häufig (Nordamerikanische Handschrift) in zwei abgesetzten Strichen geschrieben (siehe Abbildung 6.6).



(a)

(b)

Abbildung 6.5: Aufteilung einer Zahl (a) durch die Vorsegmentierung in ihre Teilbilder (b)

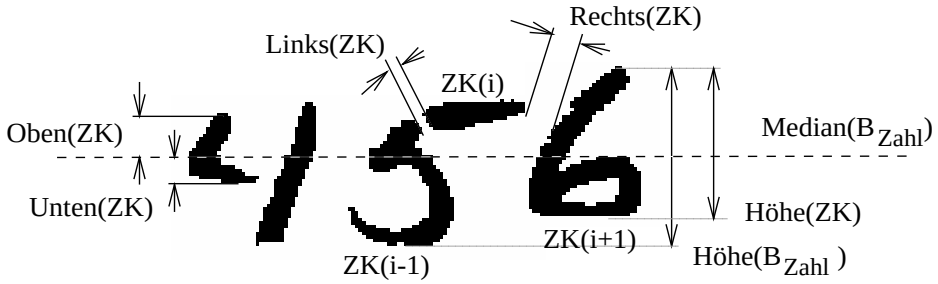


Abbildung 6.6: Definition geometrischer Grössen

Diese Beobachtungen haben zu den folgenden Kriterien für Bruchstücke geführt (die Bedeutungen der hier verwendeten geometrischen Grössen sind in Abbildung 6.6 dargestellt).

1. $\frac{Höhe(ZK)}{Höhe(B_{Zahl})} < T_{Vorsegment1}$; $T_{Vorsegment1} = 0.4$[3].
2. $\frac{max(Oben(ZK),Unten(ZK))}{min(Oben(ZK),Unten(ZK))} > T_{Vorsegment2}$; $T_{Vorsegment2} = 3.2$
3. Die ZK schneidet die horizontale, mediale Linie $Median(B_{Zahl})$ nicht.

Eine ZK wird dann als Bruchstück klassifiziert, wenn sie mindestens eine der obigen Bedingungen erfüllt. Mit der ersten Bedingungen werden zu kleine ZK's gefunden, mit der zweiten Bedingung zu asymmetrische (bezüglich der horizontalen, medialen Linie $Median(B_{Zahl})$) und mit der letzten Bedingung ZK's die so asymmetrisch

[3]Die so angegebenen Werte sind von den verwendeten Bilddaten abhängig und stellen deshalb nur Richtwerte dar. Ihre Bestimmung erfolgte in der Optimierungsphase des Zahlerkenners (siehe Kapitel 7).

liegen, das sie $Median(B_{Zahl})$ nicht mehr schneiden, wie dies für die ZK '$ZK(i)$' in Abbildung 6.6 der Fall ist.

Nach der Detektion der Bruchstücke erfolgt ihre Gruppierung. Diese wird dadurch initialisiert, dass jede ZK als Teilbild betrachtet wird. Als Bruchstücke gekennzeichnete Teilbilder werden mit ihren rechten und/oder linken Nachbarn zu neuen Teilbildern verknüpft, die wiederum darauf überprüft werden, ob sie immer noch als Bruchstücke klassifiziert werden können. Weitere Gruppierungen und Überprüfungen erfolgen iterativ, bis kein Teilbild mehr als potentielles Bruchstück eingeschätzt wird. An diesem Punkt ist die Vorsegmentierung des Bildes beendet.

Die Zuordnung eines Bruchstückes $ZK(i)$ (oder eines Teilbildes, das als Bruchstück klassifiziert wurde) zu seiner Nachbarschaft erfolgt nach der folgenden Regel[4], wobei für $T_{Vorsegment3}$ etwa 0.2 angenommen werden kann.

IF$(\frac{|Links(ZK(i))-Rechts(ZK(i))|}{Höhe(B_{Zahl})} < T_{Vorsegment3})$ THEN
  Gruppiere$(ZK(i-1), ZK(i), ZK(i+1))$
ELSE IF $(Links(ZK(i)) < Rechts(ZK(i)))$ THEN
  Gruppiere$(ZK(i-1), ZK(i))$
ELSE
  Gruppiere$(ZK(i), ZK(i+1))$

Die obige Regel stellt zuerst fest, ob eine klare Zuordnung zur rechten oder linken Nachbarschaft möglich ist. Falls die beiden Abstände $Links(ZK(i))$ und $Rechts(ZK(i))$ etwa gleich gross sind, wird $ZK(i)$ mit beiden Nachbarn verbunden, um ein später nicht mehr zu korrigierenden Segmentierungsfehler zu verhindern.

Die so gewonnenen Teilbilder werden von links nach rechts sortiert und anschliessend, wie in Abbildung 6.5 dargestellt, sequentiell dem Modul 'Detektion Ziffern' übergeben.

[4]Es wird nur die erste Iteration der Gruppierung betrachtet, deshalb wird hier nur von ZK's und nicht von Teilbildern gesprochen.

6.5 Modul 'Detektion Ziffern'

Jedes Teilbild B_{Teil} aus der Vorsegmentierung wird im Modul 'Detektion Ziffern' darauf untersucht, ob es sich um eine isolierte Ziffer oder eine Zifferngruppe handelt. Falls ein Teilbild als isolierte Ziffer akzeptiert wird, wird diese von einem Zifferner-kenner erkannt und das Resultat R_{Ziffer} dem Modul 'Kombination & Entscheidung' übergeben, andernfalls muss das Teilbild unverändert ans Modul 'Erkennung Teilbild' gesandt werden.

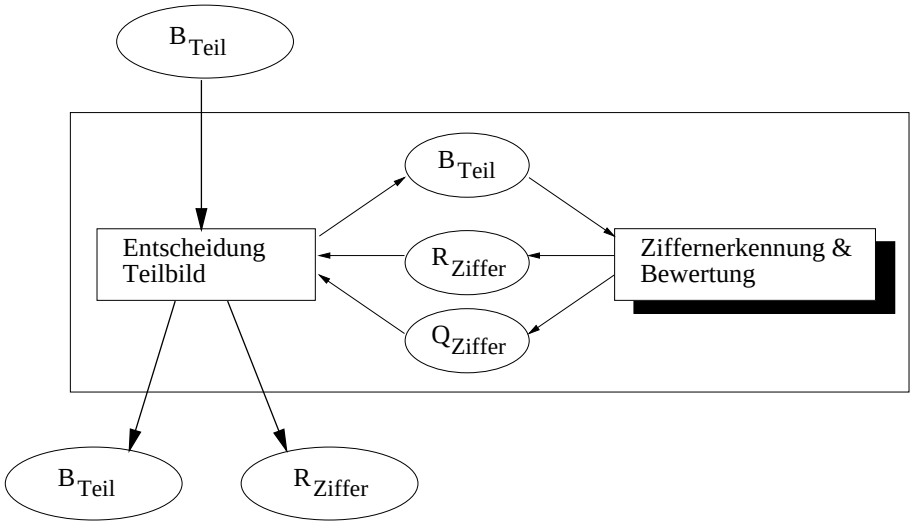


Abbildung 6.7: Aufbau des Modules zur Detektion von Ziffern

Abbildung 6.7 stellt das Modul 'Detektion Ziffer' im Überblick dar. Der Entscheid, ob es sich beim untersuchten Teilbild um eine isolierte Ziffer handelt oder nicht, wird aufgrund eines Vergleiches der berechneten Masszahl Q_{Ziffer} mit einer vordefinierten Schwelle $T_{Detektion1}$ getroffen. Die Ausgabe des Moduls 'Detektion Ziffer' besteht deshalb entweder aus dem Resultat R_{Ziffer} oder dem Teilbild B_{Teil}.

Die Entscheidung, ob es sich bei einem gegebenen Teilbild um eine isolierte Ziffer oder eine Zifferngruppe handelt, wird im Modul 'Entscheidung Teilbild' durch die folgende Regel getroffen, wobei $T_{Detektion}$ etwa bei 0.02 liegen sollte.

IF $(Q_{Ziffer}(B_{Teil}) > T_{Detektion})$ THEN
  Übergebe R_{Ziffer} dem Modul 'Kombination & Entscheidung'
ELSE
  Übergebe B_{Teil} dem Modul 'Erkennung Teilbild'

Die Masszahl $Q_{Ziffer}(B_{Teil})$ wird durch den Prozess 'Ziffernerkennung & Bewertung'

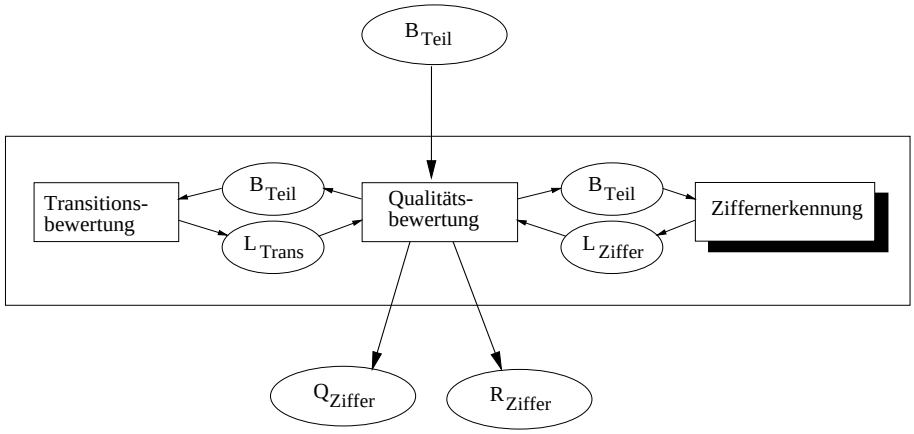


Abbildung 6.8: Aufbau des Modules zur Erkennung und Bewertung von Ziffern

berechnet, der das Resultat eines Ziffernerkenners[5] L_{Ziffer} mit einer Transitionsbewertung L_{Trans} verbindet, wie dies durch Abbildung 6.8 schematisch dargestellt ist.

Im folgenden wird zuerst kurz auf das Modul 'Ziffernerkennung' und anschliessend auf das Modul 'Qualitätsbewertung' eingegangen. Die Funktionsweise des Moduls 'Transitionsbewertung' wird am Ende dieses Abschnittes erläutert.

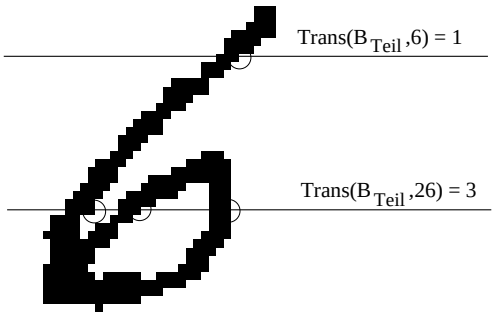


Abbildung 6.9: Bestimmung der Anzahl Transitionen für einzelne Bildzeilen

Wie in Tabelle 6.1 (links) dargestellt, liefert das Modul 'Ziffernerkennung' mit L_{Ziffer} für jede mögliche Interpretation von '0' bis '9' eine Wertung für die Korrektheit einer entsprechenden Interpretation. Die in der Tabelle angegebenen Werte wurden für das in Abbildung 6.9 dargestellte Teilbild berechnet.

[5] Der Eingesetzte Ziffernerkenner wurde ebenfalls am IAM entwickelt und verwendet eine Kombination eines kontur-basierten (siehe Kapitel 4) und eines projektions-basierten Ziffernerkenners [HB94].

Ziffer	
Klasse	Wertung
'0'	0.0646
'1'	0.0001
'2'	0.0001
'3'	0.0000
'4'	0.0006
'5'	0.0000
'6'	0.9987
'7'	0.0000
'8'	0.0012
'9'	0.0000

Trans	
Klasse	Wertung
'0'	0.9941
'1'	0.0000
'2'	0.2240
'3'	0.0025
'4'	0.1477
'5'	0.0230
'6'	0.4211
'7'	0.0187
'8'	0.6084
'9'	0.1866

Tabelle 6.1: Resultatlisten L_{Ziffer} (links) und L_{Trans} (rechts)

Im Modul 'Qualitätsbewertung' erfolgt die Kombination der Wertungen der Ziffernerkennung $q_k(B_{Teil})$ (Wertung für die Ziffernklasse k aus Tabelle L_{Ziffer}) und der Transitionsbewertung $t_k(B_{Teil})$ (analog aus Tabelle L_{Trans}) nach der folgenden Definition für $Q_{Ziffer}(B_{Teil})$

$$Q_{Ziffer}(B_{Teil}) = \frac{1}{10} \sum_{k=0}^{9} q_k(B_{Teil}) \times t_k(B_{Teil}) \tag{6.1}$$

Da es bei dieser Berechnung nicht um die korrekte Ziffernklasse geht, sondern darum, ob es sich überhaupt um eine Ziffer handelt, werden die Werte $q_k(B_{Teil}) \times t_k(B_{Teil})$ über alle Klassen aufsummiert. Für Abbildung 6.9 ergibt sich für Q_{Ziffer} der Wert 0.4856, womit dieses Teilbild als isolierte Ziffer zu interpretieren ist ($Q_{Ziffer} = 0.4856 > 0.02 = T_{Detektion}$). Das Erkennungsresultat des Ziffernerkenners R_{Ziffer} kann in diesem Fall direkt aus der Liste L_{Ziffer} entnommen werden (Resultat '6' mit der Wertung 0.9987).

Wie der Ziffernerkenner liefert auch das Modul 'Transitionsbewertung' für jede mögliche Interpretation '0' bis '9' eine Bewertung für die Korrektheit der entsprechenden Interpretation. Diese Bewertung basiert allerdings nur auf einem einzigen Merkmal des Teilbildes B_{Teil}, der mittleren Anzahl der horizontalen Transitionen. Eine horizontale Transition liegt genau dann vor, wenn (von links nach rechts) ein weisser Bildpunkt auf einen schwarzen Bildpunkt folgt.

Mit der mittleren Anzahl der horizontalen Transitionen ist der Mittelwert der Zahl der horzontalen Transitionen pro Bildzeile gemeint. In Abbildung 6.9 ist die Bestimmung der Anzahl Transitionen der Bildzeile 6 und 26 (von oben nach unten) dargestellt. Mit der Höhe dieses Teilbildes von 38 Bildpunkten ergibt sich hier für die mittlere Anzahl der horizontalen Transitionen $Trans(B_{Teil})$ der Wert 1.55.

Klasse	m	s
'0'	1.5513	0.1273
'1'	0.9393	0.0220
'2'	1.2713	0.1621
'3'	1.1823	0.0861
'4'	1.3353	0.1049
'5'	1.1747	0.1168
'6'	1.3167	0.2051
'7'	1.1293	0.1265
'8'	1.4420	0.1490
'9'	1.2840	0.1424

Tabelle 6.2: Mittlere Anzahlen der Transitionen und deren Standardabweichungen

Da die Ziffern einer Zahl in der Regel etwa dieselbe Höhe aufweisen und nebeneinander stehen, steigt auch der Wert $Trans(B_{Teil})$ mit der Anzahl im Teilbild B_{Teil} enthaltenen Ziffern an, wodurch sich Ziffern in der Praxis zuverlässig von Zifferngruppen unterscheiden lassen.

Die Werte für die mittlere Anzahlen der Transitionen weichen ausserdem für die verschiedenen Ziffernklassen voneinander ab, wie Messungen von mehreren hundert Beispielen ergeben haben. Die Resultate dieser Messungen sind in der Tabelle 6.2 dargestellt, wobei die Spalte 'm' den gemittelten Wert $Trans(B_{Teil})$ für jede Ziffernklasse k ($\hat{\mu}_k$) und die Spalte 's' die zugehörige, empirische Standardabweichung enthält ($\hat{\sigma}_k$).

Dieser Sachverhalt erlaubt die Konstruktion einer klassenspezifischen "Transitionsbewertung", da anhand einer konkreten Messung von $Trans(B_{Teil})$ die Wahrscheinlichkeit für die Zugehörigkeit des Teilbildes zur Klasse k berechnet werden kann. Die Berechnung der Wertungen $t_k(B_{Teil})$ für die Liste L_{Trans} erfolgt durch die Auswertung der folgenden Gleichung, wobei $erf()$ der Fehlerfunktion der Normalverteilung entspricht[6].

$$t_k(B_{Teil}) = 1 - erf(\frac{|Trans(B_{Teil}) - \hat{\mu}_k|}{\hat{\sigma}_k}) \tag{6.2}$$

Durch die Symmetrien $erf(0) = 0, erf(\infty) = 1$ und $erf(-x) = -erf(x)$ wird in obiger Gleichung eine Wertung $t_k \in [0,1]$ erreicht, die bei schlechter Übereinstimmung von $Trans()$ mit der Verteilung der entsprechenden Ziffernklasse in einem Wert nahe bei 0.0 resultiert. Bei guter Übereinstimmung liegt dieser Wert nahe bei 1.0. Für Abbildung 6.9 ist die aus diesen Berechnungen resultierende Liste L_{Trans} in

[6]Die Fehlerfuntion ist durch $erf(x) = 2/sqrt(\pi) \int (t = 0, t = x) e^{-t^2} dt$ definiert und in der mathematischen Standard-Bibliothek des verwendeten C-Compilers vorhanden.

Tabelle 6.1 wiedergegeben, wobei die Spalte 'Wertung' die Werte t_k enthält. Trotz der Einfachheit dieses "Erkenners" erreicht die korrekte Klasse '6' immerhin die drittbeste Wertung.

6.6 Modul 'Erkennung Teilbild'

Die Aufgabe des Moduls 'Erkennung Teilbild' besteht in der Erkennung von Teilbildern, die Zifferngruppen beinhalten. Um auch Zifferngruppen erkennen zu können, die sich berührende Ziffern enthalten oder aus Ziffern bestehen, die aus vielem ZK's zusammengesetzt sind, wird in diesem Modul ein segmentierungsfreies Verfahren eingesetzt, das ursprünglich durch H. Nishida und S. Mori in [NM92] beschrieben wurde. Abbildung 6.10 stellt dieses Modul im Überblick dar.

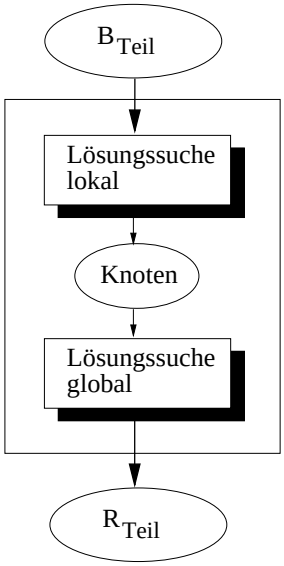


Abbildung 6.10: Aufbau des Modules zur Erkennung eines Teilbildes

Im ersten Schritt 'Lösungssuche lokal' wird ausgehend vom zu erkennenden Teilbild B_{Teil} eine Menge potentieller Ziffernbilder generiert, die als Knoten eines Graphen interpretiert werden. Im zweiten Schritt 'Lösungssuche global' wird aus dieser Menge von Knoten eine global optimale und konsistente Teilmenge bestimmt, und daraus die optimalen Segmentierungsstellen und das Erkennungsresultat R_{Teil} des Moduls abgeleitet.

In Abbildung 6.11 ist ein Beispiel eines Eingabebildes B_{Teil} und der zugehörigen Ausgabe R_{Teil} für das Modul 'Erkennung Teilbild' dargestellt.

Die beiden Module 'Lösungssuche lokal' und 'Lösungssuche global' sind in den beiden folgenden Unterabschnitten ausführlicher beschrieben. Im dritten und letzten Unterabschnitt wird das Konzept des "Dummy- Segmentes" eingeführt, das die Unterdrückung von Bildteilen ermöglicht, die keinen Ziffern des zu erkennenden Teilbildes zugeordnet werden können und mit üblichen Filtertechniken nicht zu löschen sind.

Ziffer	Resultat	Wertung
1	'4'	1.000
2	'0'	1.000
3	'9'	0.996

Abbildung 6.11: Beispiel eines Teilbildes B_{Teil} (links) mit zugehörigem Resulat R_{Teil} des Teilbilderkenners

6.6.1 Lösungssuche lokal

Das Modul 'Lösungssuche lokal' zerlegt ein Teilbild in Ziffernbilder welche als Knoten dem Modul 'Lösungssuche global' übergeben werden. Die Ziffernbilder repräsentieren die potentiellen, isolierten Ziffern, welche erkannt, bewertet und in einer Tabelle als Knoten gespeichert werden. Die Funktionsweise dieses Prozesses ist in Abbildung 6.12 wiedergegeben. Das Modul 'Ziffernerkennung & Bewertung' wurde bereits im Abschnitt 6.5 beschrieben und ist in Abbildung 6.8 dargestellt.

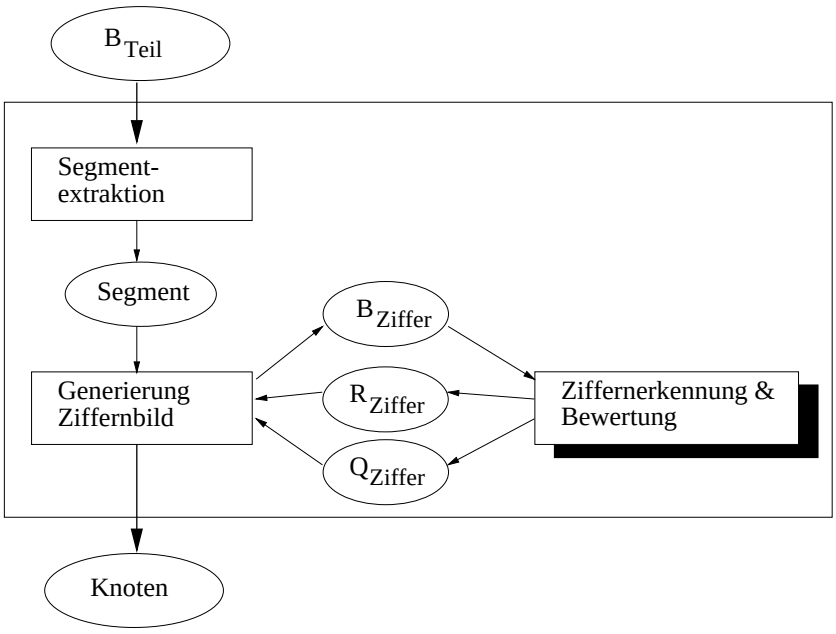


Abbildung 6.12: Aufbau des Modules zur lokalen Lösungssuche

In einem ersten Schritt 'Segmentextraktion' werden alle Segmente des Teilbildes extrahiert und anschliessend von links nach rechts sortiert. Segmente sind dadurch

gekennzeichnet, dass sie bei singulären Stellen[7] beginnen und enden. Zur Extrahierung und Sortierung der Segmente werden die folgenden Schritte durchgefürt.

1. Das Teilbild wird verdünnt.
2. Singuläre Punkte werden detektiert und gelöscht. Sehr kurze Linien zwischen zwei singulären Punkten werden ebenfalls entfernt.
3. Die resultierenden ZK's werden unterschiedlich gekennzeichnet.
4. Die gekennzeichneten ZK's werden innerhalb der ursprünglichen ZK's expandiert.
5. Die resultierenden, unterschiedlich gekennzeichneten Regionen entsprechen den gesuchten Segmenten, die noch von links nach rechts sortiert werden.

Die obigen Schritte sollen anhand eines Beispieles in Abbildung 6.13 illustriert werden.

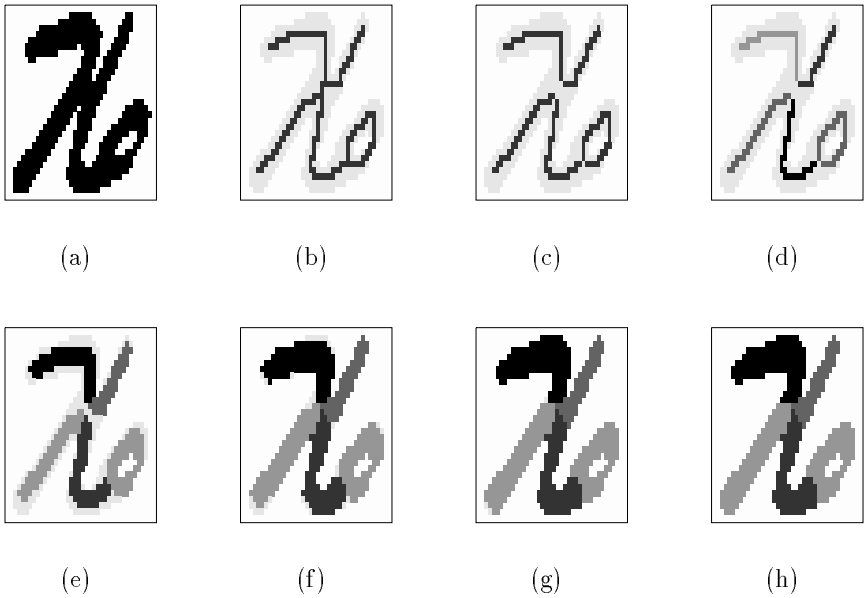


Abbildung 6.13: Extraktion der Segmente eines Teilbildes

Bild (a) stellt das ursprüngliche Teilbild dar, das eine ZK enthält, welche die Zahl '76' repräsentiert. Bild (b) zeigt das verdünnte Teilbild des zweiten Extraktionsschrittes,

[7] Mit singulären Stellen werden Endpunkte, Verzweigungen oder Kreuzungen bezeichnet. Jede singuläre Stelle des Teilbildes kann anschliessend als potentielle Segmentierungsstelle interpretiert werden.

Bild (c) das resultierende Teilbild nach dem Entfernen singulärer Punkte und kurzer Linien. Bild (d) präsentiert die resultierenden, durch unterschiedliche Graustufen gekennzeichneten ZK's. In der zweiten Zeile der Abbildung 6.13 ist die Expansion der gekennzeichneten Regionen bis zur Abdeckung der gesamten, ursprünglichen ZK dargestellt. Das letzte Bild (h) enthält die fünf resultierenden Segmente.

Im zweiten Schritt 'Generierung Ziffernbild' des Moduls 'Lösungssuche lokal' werden anhand der extrahierten Segmente potentielle Ziffernbilder B_{Ziffer} konstruiert, wobei ein Ziffernbild durch die Auswahl einer Teilmenge aller extrahierten Segmente ensteht. Das Resultat dieses Prozesses ist in der Abbildung 6.14 dargestellt.

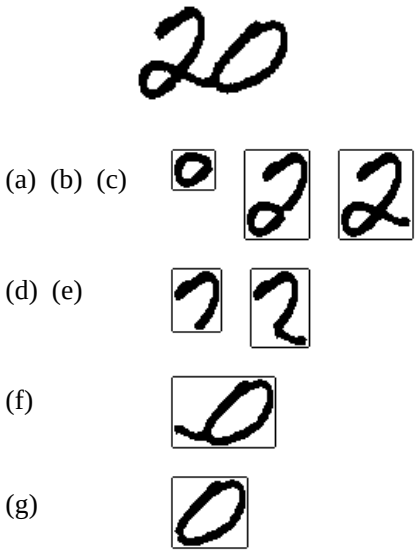


Abbildung 6.14: Aufteilung eines Teilbildes in potentielle Zifferbilder (a) ... (g)

Falls die Anzahl der extrahierten Segmente N_S beträgt, existieren $2^{N_S} - 1$ Möglichkeiten, eine Teilmenge von Segmenten zu bestimmen, woraus sich eine exponentielle Komplexität dieses Schrittes ergeben würde. Da ein grosser Anteil solch zufällig zusammengesetzter Ziffernbilder unsinnig wäre, kann die Auswahl der Teilmengen durch geometrische Einschränkungen auf quadratische Komplexität reduziert werden.

Die Auswahl der Teilmengen läuft nach der folgenden Regel ab: Zu jedem Segment $Segment(i), i = 1, \ldots, N_S$ werden $N_Z = N_S - i + 1$ Ziffernbilder $B_{Ziffer}(i, i)$ bis $B_{Ziffer}(i, N_S)$ konstruiert, welche durch die Vereinigung der entsprechenden Segmente entstehen $\{\bigcup_{j=i}^{k} Segment(j); i \leq k \leq N_S\}$. Um als Knoten zu gelten und dem Modul 'Lösungssuche Global' somit als Input übergeben zu werden, müssen diese generierten Ziffernbilder zusätzlich den folgenden beiden Bedingungen genügen:

1. $\frac{Höhe(B_{Ziffer})}{Höhe(B_{Teil})} > T_{Teilerk1}; T_{Teilerk1} = 0.3$
2. $\frac{Breite(B_{Ziffer})}{Höhe(B_{Ziffer})} < T_{Teilerk2}; T_{Teilerk2} = 4.0$

Durch die erste Bedingung werden zu kleine Ziffernbilder unterdrückt und die zweite Bedingung eliminiert Fälle, welche ein für Ziffern unsinniges Verhältnis von Breite zu Höhe aufweisen.

Die verbleibenden Ziffernbilder werden anschliessend durch das Modul 'Ziffernerkennung & Bewertung' erkannt (R_{Ziffer}) und bewertet (Q_{Ziffer}). Anschliessend wird die Eignung des Teilbildes, eine isolierte Ziffer darzustellen, untersucht. Diese ist durch eine Wertung Q_{Knoten} quantifizierbar, die sich aus den Faktoren Q_{Ziffer} und $Q_{Einbettung}$ zusammensetzt.

$$Q_{Knoten}(B_{Ziffer}) = Q_{Ziffer}(B_{Ziffer}) \times Q_{Einbettung}(B_{Ziffer}) \tag{6.3}$$

Der Faktor $Q_{Einbettung}$ stellt ein Mass für die Qualität der Einbettung des Ziffernbildes im Teilbild dar und wird wie folgt bestimmt (siehe auch Abbildung 6.6):

$$Q_{Einbettung}(B_{Ziffer}) = \frac{1}{1 + (\frac{Median(B_{Ziffer}) - Median(B_{Teil})}{Höhe(B_{Teil})})^2} \tag{6.4}$$

In Tabelle 6.3 sind Informationen zu den Teilbildern der Abbildung 6.14 zusammengefasst. Die Spalte 'Knoten' enthält die Bezeichnung des Knotens. Die Spalten 'S1' bis 'S4' geben Aufschluss darüber, welche Segmente zur Generierung des zugehörigen Teilbildes verwendet wurden. Die Spalten 'Resultat' und 'Wertung' enhalten die entsprechenden Angaben aus R_{Ziffer} und die Spalte 'Knoten-Wertung' die berechneten Zahlen für Q_{Knoten}.

Knoten	S1	S2	S3	S4	Resultat	Wertung	Knoten-Wertung
(a)	X				'0'	1.000	0.774
(b)	X	X			'2'	0.643	0.640
(c)	X	X	X		'2'	1.000	1.000
(d)		X			'7'	0.998	0.798
(e)		X	X		'2'	0.992	0.942
(f)			X	X	'0'	0.430	0.409
(g)				X	'0'	0.997	0.947

Tabelle 6.3: Durch 'Lösungssuche lokal' bestimmte Knoten

6.6.2 Lösungssuche global

Das Modul 'Lösungssuche global' berechnet anhand der extrahierten und bewerteten Knoten eine konsistente und optimale Interpretationen für das zu erkennende Teilbild B_{Teil}. Die Berechnung der Lösung erfolgt durch eine Graphsuche[8], wie dies in der Übersicht dieses Moduls in Abbildung 6.15 dargestellt ist.

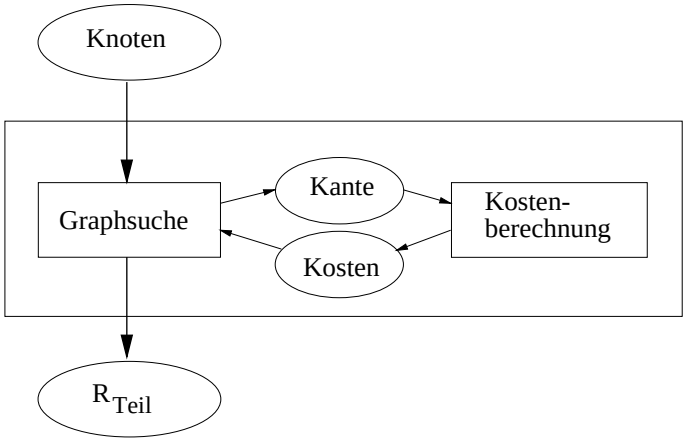


Abbildung 6.15: Aufbau des Modules zur globalen Lösungssuche

Im Modul 'Graphsuche' wird ein gerichteter und gewichteter Graph durch die Graphsuche dynamisch aufgebaut, wobei die Kosten der untersuchten Kanten laufend durch das Modul 'Kostenberechnung' geliefert werden.

Die Knoten des Graphen repräsentieren potentielle, isolierte Ziffern. Um die Graphsuche bei genau einem Knoten beginnen und genau einem Knoten abschliessen zu können, werden zusätzlich die künstlichen Knoten 'Start' und 'Ziel' eingeführt (siehe Abbildung 6.17). Durch eine gerichtete Kante des Graphen wird jeweils die Reihenfolge der Interpretation zweier Knoten festgelegt. Die Gewichtung der Kanten wird von einer Kostenfunktion berechnet, welche die "Natürlichkeit" der Abfolge der zwei untersuchten Knoten berücksichtigt. Jeder Pfad vom Knoten 'Start' zum Knoten 'Ziel' (Lösungspfad) stellt durch die enthaltenen Knoten und die durch die gerichteten Kanten gegebene Reihenfolge eine Interpretation des zu erkennenden Teilbildes dar.

Im folgenden werden zuerst die Bedingungen für eine konsistente Interpretation des Teilbildes behandelt und anschliessend die Optimalität einer solchen Interpretation

[8]Zur Graphsuche wurde mit den Algorithmus A* eine Breitensuche implementiert, wie sie im Vorlesungs-Skript 'KI' beschrieben ist [Bun92].

definiert. Die konsistenten Interpretationen des zu erkennenden Teilbildes werden durch die folgenden Bedingungen an den Lösungspfad definiert:

1. Der Lösungspfad muss vom Knoten 'Start' ausgehend unmittelbar über einen Knoten führen, der das erste extrahierte Segment des Teilbildes beinhaltet.
2. Der Lösungspfad muss unmittelbar vor dem letzten Knoten 'Ziel' über einen Knoten führen, der das letzte extrahierte Segment beinhaltet.
3. Eine vom Knoten i ausgehende Kante darf nicht wieder zum Knoten i führen.
4. Eine Kante vom Knoten i zum Knoten j $(i \neq j)$ existiert genau dann, wenn das erste Segment des Knotens j direkt auf das letzte Segment des Knotens i folgt oder zu diesem identisch ist.

Die ersten beiden Bedingungen legen den Beginn und das Ende eines Lösungspfades fest. Für das Beispiel in Abbildung 6.14, beziehungsweise Tabelle 6.3, würde dies bedeuten, dass ein Lösungspfad vom Knoten 'Start' über die Knoten (a),(b) oder (c) führen und über die Knoten (f) oder (g) zum Knoten 'Ziel' laufen muss. Die dritte Bedingung verhindert, dass ein Knoten mehr als einmal interpretiert werden kann. Die vierte Bedingung legt fest, dass die Interpretation des Teilbildes von links nach rechts erfolgen muss und stellt sicher, dass alle extrahierten Segmente zur Interpretation des Teilbildes beitragen. Ein Lösungspfad ((a),(f)) ist aufgrund der Tabelle 6.3 nicht zulässig, da er das zweite Segment 'S2' (Knoten (d)) zwischen den beiden Knoten (a) und (f) nicht berücksichtigt. Auch der Lösungspfad ((c),(e),(g)) ist ungültig, da sich die beiden Knoten (c) und (e) in den zwei Segmenten 'S2' und 'S3' überschneiden. Einige der möglichen Lösungspfade sind in Abbildung 6.17 graphisch dargestellt.

Um nicht nur eine konsistente sondern auch eine optimale Interpretation für das Teilbild B_{Teil} finden zu können, werden den Kanten des Graphen Kosten C_{Kante} zugewiesen. Eine optimale Interpretation wird somit durch einen Lösungspfad erreicht, der minimale Kosten aufweist. Die Definition der Kosten muss deshalb eine korrekte Interpretation des Teilbildes zu minimalen Kosten erlauben.

Die Kosten einer Kante $C_{Kante}(i,j)$ zwischen den beiden Knoten i und j setzen sich aus den Kosten für die beiden Knoten $C_{Knoten}(i)$ und $C_{Knoten}(j)$ sowie den Kosten für die Verbindung $C_{Verbindung}(i,j)$ zusammen, wobei der Parameter $W_{Knoten} = 0.9$ die Gewichtung der Knoten-Kosten gegenüber den Verbindungskosten bestimmt:

$$C_{Kante}(i,j) = W_{Knoten} C_{Knoten}(i,j) + (1 - W_{Knoten}) C_{Verbindung}(i,j) \quad (6.5)$$

$$C_{Knoten}(i,j) = C_{Knoten}(i) + C_{Knoten}(j) \quad (6.6)$$

$$C_{Verbindung}(i,j) = C_{ah}(i,j) + C_{av}(i,j) \quad (6.7)$$

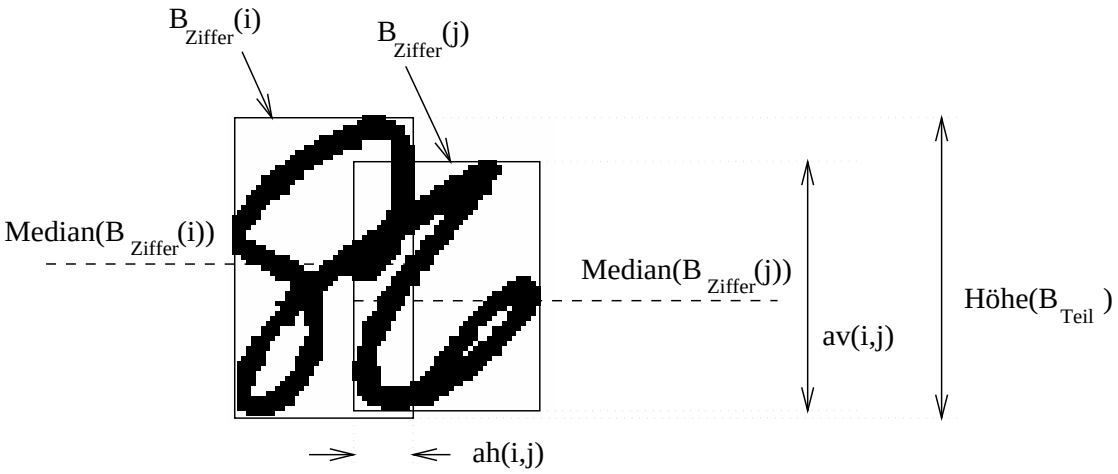


Abbildung 6.16: Beziehungen zwischen benachbarten Zifferbildern

Die Kosten für die Knoten i und j werden wie folgt aus den zugehörigen Massen Q_{Ziffer} berechnet (siehe auch Abschnitt 6.5).

$$C_{Knoten}(i) \quad = \quad \frac{100}{1 + Q_{Ziffer}(B_{Ziffer}(i))} \tag{6.8}$$

Die Kosten für die Verbindung $C_{Verbindung}(i,j)$ zweier Knoten ergeben sich durch die Summe der Kosten für die horizontale Abdeckung $C_{ah}(i,j)$ der beiden Ziffernbilder und den Kosten für die vertikale Abdeckung $C_{av}(i,j)$ (siehe Abbildung 6.16):

$$C_{ah}(i,j) = 1 + 100\frac{ah(i,j)}{min(Breite(B_{Ziffer}(i)), Breite(B_{Ziffer}(j)))} \tag{6.9}$$

$$C_{av}(i,j) = 1 + 100[\frac{Median(B_{Ziffer}(i)) - Median(B_{Ziffer}(j))}{Höhe(B_{Zahl})}]^2 \tag{6.10}$$

Die so definierten Kosten einer Kante werden genau dann minimal, wenn die verbundenen Knoten mit hoher Wahrscheinlichkeit isolierte Ziffern darstellen, sich nicht horizontal überlappen und beide Knoten dieselbe vertikale Ausrichtung besitzen. Ein Beispiel für eine solche Graphsuche ist in Abbildung 6.17 dargestellt, wobei die Knoten und deren Bezeichnungen mit Abbildung 6.14 übereinstimmen.

Die Zahlen neben den Kanten geben die berechneten Kosten wieder, wobei die Knoten 'Start' und 'Ziel' mit fixen Knoten- und Verbindungskosten zu den entsprechenden Kanten beitragen. Der günstigste Lösungspfad führt vom Knoten (Start) über die Knoten (c) und (g) zum Knoten (Ziel).

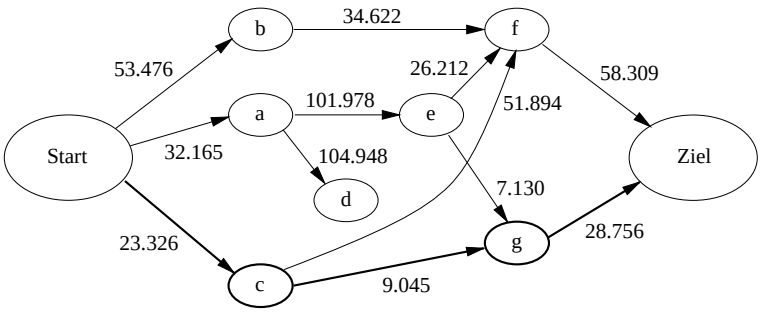


Abbildung 6.17: Darstellung der Graphsuche im Modul 'Lösungssuche global'

Anhand dieses Lösungspfades und Tabelle 6.3 kann das Teilresultat R_{Teil} zusammengesetzt werden, das in diesem Beispiel durch die '2' mit Wertung 1.000 gefolgt von der '0' mit Wertung 0.997 gegeben ist (ein weiteres Beispiel wurde bereits in Abbildung 6.11 gegeben).

6.6.3 Konzept Dummy-Segment

Das Konzept des Dummy-Segmentes erlaubt die Behandlung von Spezialfällen bei der Interpretation von Teilbildern. Einzelne Segmente, die nicht zu isolierten Ziffern passen, wie Ligaturen oder Bindestriche, können so unterdrückt werden. Eine Ligatur verbinden zwei Ziffern durch einen zusätzlichen Linienzug, wie dies beim Ziffernpaar '00' in Abbildung 6.18 der Fall ist. Bindestriche treten beispielsweise in den 9-stelligen Postleitzahlen der CEDAR Datenbank auf.

(a) (b)

Abbildung 6.18: Beispiel für Teilbilder mit Ligatur (a) und Bindestrich (b)

Um dieses Konzept im beschriebenen Teilbilderkenner einzubauen, sind zwei Modifikationen notwendig. Als erste Modifikation werden im Modul 'Lösungsuche lokal' die Ziffernbilder darauf untersucht, ob sie potentielle Dummy-Segmente darstellen könnten.

Potentielle Dummy-Segmente müssen die beiden folgenden Bedingungen erfüllen.

1. das Ziffernbild besteht aus einem einzigen Segment
2. $\frac{Breite(B_{Ziffer})}{Höhe(B_{Ziffer})} > T_{dummy}; T_{dummy} = 5$

Die erste Bedinung schränkt die Suche auf einfache Ziffernbilder ein und die zweite Bedingung fordert zusätzlich, dass potentielle Dummy-Segmente bezüglich deren Höhe sehr breit sein müssen. Diese beiden Bedingungen sind so gewählt, dass die meisten Ligaturen oder Trennstriche als potentielle Dummy-Segmente markiert werden. Richtige Ziffern bestehen oft aus mehreren Segmenten oder haben zumindest ein kleineres Verhältnis $Breite(B_{Ziffer})/Höhe(B_{Ziffer})$.

Im Module 'Lösungsuche global' erfolgt eine zweite Modifikation. Die vierte Bedingung an Lösungspfade wird so ergänzt, dass eine Verbindung zweier Knoten i und j auch dann möglich ist, falls zwischen den beiden Knoten ein als Dummy-Segment markiertes Segment "verschluckt" wird. So ist es möglich, dass eine Ligatur oder ein Bindestrich nicht zur Interpretation des Teilbildes verwendet werden muss.

Um zu verhindern, dass durch dieses Konzept zuviele Segmente bei der Interpretation des Teilbildes unterdrückt werden, ist es notwendig, die Kosten von Dummy-Verbindungen gegenüber normalen Verbindungen künstlich zu erhöhen. Die Kosten für eine Kante von einem Knoten i über das Dummy-Symbol λ zum Knoten j sind wie folgt definiert:

$$C_{Kante}(i, \lambda, j) = W_{Dummy1} + W_{Dummy2} C_{Kante}(i, j) \tag{6.11}$$

Durch $W_{Dummy1} = 30$ wird erreicht, dass eine solche Kante nie kostenlos sein wird und durch $W_{Dummy2} = 2$, dass diese Kosten zusätzlich (kantenabhängig) erhöht werden. Dummy-Verbindungen können durch diese Definitionen ohne Veränderung der Graphsuche eingeführt werden.

6.7 Modul 'Kombination und Entscheidung'

Dieser Abschnitt ist in zwei Unterabschnitte eingeteilt. Im ersten Unterabschnitt wird auf die Problematik von Entscheidungsregeln zur Rückweisung von Beispielen eingegangen und der zweite Unterabschnitt beschäftigt sich mit dem Modul 'Kombination und Entscheidung'.

6.7.1 Problematik von Entscheidungsregeln

Die Aufgabe einer Rückweisungsregel besteht darin, eine gegebene Fehlerrate[9] mit einer minimalen Rückweisungsrate zu erreichen, beziehungsweise für eine gegebene Rückweisungsrate eine minimale Fehlerrate zu erzielen. Durch die Zuweisung einer Fehlerrate zu einer bestimmten Rückweisungsrate lässt sich dieser Zusammenhang als Funktion $R_{Fehler}(R_{Rück})$ darstellen.

Für eine gegebene Rückweisungsregel, die beliebige Rückweisungsraten $\in [0,1]$ erlaubt, kann eine charakteristische Funktion $f_{geg}(t)$ experimentell bestimmt werden, die zu jeder Rückweisungsrate t die zugehörige Fehlerrate f_{geg} liefert.

Für die Fehlerrate $f_{geg}(0) = \lambda$ erreicht eine optimale Rückweisungsregel für $t = \lambda$ die Fehlerrate 0. Ein Erkennungssystem mit einer solchen Rückweisungsregel weist bis zur Rückweisungsrate λ ausschliesslich Beispiele zurück, die nicht korrekt erkannt worden wären. Diese optimale Rückweisungsregel lässt sich durch die folgende Funktion charakterisieren.

$$f_{opt}(t) = \begin{cases} t \leq \lambda: & \lambda - t \\ t > \lambda: & 0 \end{cases} \tag{6.12}$$

Die triviale Rückweisungsregel weist zu erkennende Beispiele mit der Wahrscheinlichkeit der gegebenen Rückweisungsrate zurück, ohne irgendwelche zusätzlichen Informationen zu berücksichtigen. Dadurch wird eine Fehlerrate von 0 erst bei der Rückweisungsrate von 1 erreicht (was 100% entspricht). Das Ergebnis der Anwendung dieser Regel kann somit durch die folgende Funktion charakterisiert werden.

$$f_{triv}(t) = \lambda(1 - t) \tag{6.13}$$

Die Abbildung 6.19 veranschaulicht diese beiden Funktionen f_{opt} und f_{triv} zusammen mit einer fiktiven Messung $f_{geg}(t)$ für eine konkrete Rückweisungsregel.

Die Qualität $Q_{Regel} \in [0,1]$ einer gegebenen Rückweisungsregel lässt sich somit durch einen Vergleich mit der optimalen und der trivialen Rückweisungsregel bewerten. Die

[9] siehe Abschnitt 3.2.

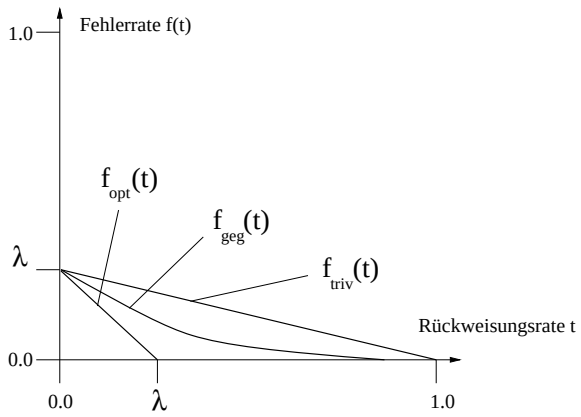


Abbildung 6.19: Resultate verschiedener Rückweisungsregeln

folgende Gleichung kann als Definition für das Mass Q_{Regel} verwendet werden.

$$Q_{Regel} = 1 - \frac{\int_0^1 f_{geg}(t) - f_{opt}(t)\,dt}{\int_0^1 f_{triv}(t) - f_{opt}(t)\,dt} \tag{6.14}$$

Durch $Q_{Regel} = 1$ wird somit ausgedrückt, dass das Rückweisungsverhalten der untersuchten Regel optimal ist. Für $Q_{Regel} = 0$ müsste man folgern, dass die untersuchte Regel nicht besser ist, als eine zufällige Rückweisung von Beispielen.

6.7.2 Implementation des Moduls

Im Modul 'Kombination und Entscheidung' werden die Erkennungsresultate der beiden Module 'Detektion Ziffern' und 'Erkennung Teilbild' zu einem Zwischenresultat kombiniert. Anschliessend wird aufgrund dieses Ergebnisses entschieden, ob das zu erkennende Bild zurückgewiesen werden muss oder das berechnete Resultat ausgegeben werden kann. Eine Übersicht zu diesem Modul ist in Abbildung 6.20 wiedergegeben.

In Schritt 'Resultatkombination' werden die Resultate R_{Ziffer} und R_{Teil} der Module 'Detektion Ziffer' und 'Erkennung Teilbild' im Zwischenresultat R_{Zahl} zusammengefasst. Die einzelnen Resultate werden dazu in der Reihenfolge (von links nach rechts) der zugehörigen Teilbilder sortiert. Neben den Angaben der Ziffernklassen und Wertungen wird ausserdem die Information zur Herkunft des entsprechenden Ergebnisses ins Zwischenresultat R_{Zahl} aufgenommen.

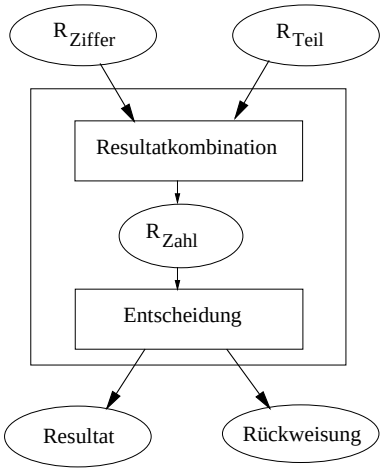


Abbildung 6.20: Aufbau des Modules 'Kombination und Entscheidung'

Für das Beispiel der Abbildung 6.5 ist die Kombination der einzelnen Resultate in Tabelle 6.4 wiedergegeben, wobei die Spalte 'Ziffer' der Postition und die Spalte 'Modul' der Herkunft des Resultates entsprechen.

Ziffer	Modul	Klasse	Wertung
1	Ziffer	'5'	0.917
2	Ziffer	'2'	0.999
3	Ziffer	'7'	0.998
4	Teilbild	'5'	0.993
5	Teilbild	'2'	0.983
6	Ziffer	'1'	0.995

Tabelle 6.4: Beispiel für ein Zwischenresultat R_{Zahl} des Zahlenerkenners

Im zweiten Schritt 'Entscheidung' wird die Entscheidung getroffen, die berechnete Interpretation (Fall 'Resultat') auszugeben, oder das zu erkennende Bild zurückzuweisen (Fall 'Rückweisung'). Die Aufgabe der Entscheidungsregel besteht darin, die Fehlerrate des Zahlerkenners für eine gegebene Rückweisungsrate zu minimieren. Um dieses Ziel zu erreichen, dürfen Bilder nur dann zurückgewiesen werden, wenn aufgrund von Aufzeichnungen während des Erkennungsvorganges die Korrektheit der berechneten Interpretation in Frage gestellt werden muss.

Die Korrektheitsbewertung der berechneten Interpretation erfolgt anhand eines Masses Q_{Zahl}, das wie folgt aus dem Zwischenresultat R_{Zahl} gewonnen werden kann.

$$Q_{Zahl} = \min_i \; Wertung(R_{Zahl}(i)) \tag{6.15}$$

Die Definition von Q_{Zahl} orientiert sich an der Bewertung von Erkennungsresultaten (siehe Abschnitt 3.2). *Ein Beispiel gilt genau dann als richtig erkannt, wenn das Resultat des Erkennungssystems mit dem Wahrheitswert des Beispieles übereinstimmt.* Wenn sich das Resultat des Erkennungssystems auch nur in einer Ziffer vom Wahrheitswert des betreffenden Beispieles unterscheidet, gilt dies als Erkennungsfehler.

Das Mass Q_{Zahl} wird also durch die tiefste Wertung über alle in R_{Zahl} enthaltenen Ziffern definiert, womit sich für das Beispiel in Tabelle 6.4 für Q_{Zahl} den Wert 0.917 ergeben würde. Anschaulich gesprochen, entspricht Q_{Zahl} der Stärke des schwächsten Gliedes (Ziffer) in der Kette (Zahl).

Experimente haben gezeigt, dass es anhand des Erkennungsvorganges möglich ist, "einfache" und "komplexe" Bilder zu unterscheiden, wobei Bilder genau dann als "einfach" bezeichnet werden, wenn ihre Erkennung den Einsatz des Moduls 'Erkennung Teilbild' nicht erfordert. Solche Bilder können im Gegensatz zu "komplexen" Bilder mit hoher Zuverlässigkeit erkannt werden. "Komplexe" Bilder sind in der Regel schwieriger zu segmentieren und erfordern deshalb den Einsatz des Moduls 'Erkennung Teilbild', was zur Folge hat, dass ihre Interpretation weniger zuverlässig ist.

Aufgrund dieser Beobachtung berücksichtigt die Entscheidungsregel nicht nur die Bewertung Q_{Zahl} des Zwischenresultates, sondern auch die Herkunft der einzelnen Ziffernresultate. Um die Fehlerraten des Systems für beliebige Rückweisungsraten messen zu können, lässt sich die Entscheidungsregel ausserdem durch einen Parameter $t \in [0, 1]$ beeinflussen.

IF (Modul 'Erkennung Teilbild' benutzt) THEN
  IF ($Q_{Zahl} < T_{Entscheidung1}(t)$) THEN
    Rückweisung
  ELSE
    Ausgabe des Resultates
ELSE
  IF ($Q_{Zahl} < T_{Entscheidung2}(t)$) THEN
    Rückweisung
  ELSE
    Ausgabe des Resultates

Der Verlauf der beiden Schwellwertfunktionen $T_{Entscheidung1}(t)$ und $T_{Entscheidung2}(t)$ ist in Abbildung 6.21 wiedergegeben. Für $t = A$ ergeben sich die beiden Schwellwerte $T_{Entscheidung1}(t) = B$ und $T_{Entscheidung2}(t) = C$. Bei der Erkennung "komplexer" Bilder werden somit höhere Anforderungen an die Wertung Q_{Zahl} gestellt als bei "einfachen" Bildern.

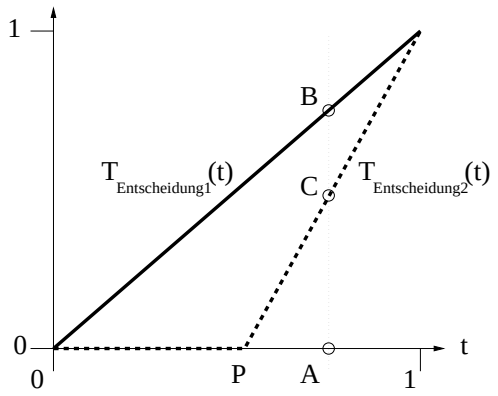


Abbildung 6.21: Die Schwellwertfunktionen $T_{Entscheidung1}(t)$ und $T_{Entscheidung2}(t)$

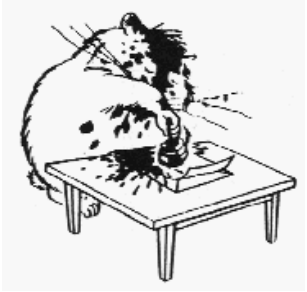
Falls t kleiner als $P = 0.85$ gewählt wird[10], ergibt sich für die Funktion $T_{Entscheidung2}(t)$ der Wert 0.0 und der Zahlenerkenner kann nur noch Bilder zurückweisen, zu deren Erkennung das Modul 'Erkennung Teilbild' eingesetzt wurde. Für den Extremfall $t = 0.0$ nehmen beide Schwellwerte den Wert 0.0 und das Erkennungssystem muss alle Bilder interpretieren. Der andere Extremwert des Parameters liegt bei 1.0 was dazu führt, dass der Zahlenerkenner alle Bilder zurückweist, da die Wertung Q_{Zahl} nie grösser als 1.0 sein wird.

Wenn der Zahlerkenner das zu erkennende Bild akzeptiert, wird aus dem Zwischenresultat R_{Ziffer} schliesslich das Endergebnis der ganzen Erkennung, in der Form von Tabelle 6.5, zusammengesetzt und ausgegeben.

Resultat	Wertung
'527521'	0.917

Tabelle 6.5: Ein Resultat des Zahlenerkenners

[10] Der Wert P muss in der Optimierungsphase des Zahlenerkenners experimentell bestimmt werden und ist (wie der Parameter t) nicht vom zu erkennenden Bild abhängig.



Kapitel 7

Experimente und Resultate

In diesem Kapitel werden einige der mit dem Zahlenerkenner durchgeführten Experimente präsentiert. Die ausgewählten Experimente sollen nicht nur die kontinuierlichen Verbesserungen und Erweiterungen beschreiben, sondern auch erfolglose Veränderungen des Erkennungssystems dokumentieren.

Der erste Abschnitt 7.1 stellt die Optimierungsphase (oder das Training) des Zahlenerkenners vor. Eine Analyse des Einsatzes der einzelnen Module des Zahlenerkenners und der Erkennungsfehler erfolgt im Abschnitt 7.2. Experimente zur Entwicklung der Rückweisungsregel werden im Abschnitt 7.3 behandelt und im Abschnitt 7.4 werden die Resultate einiger zusätzlicher Messungen besprochen. Die beiden letzten Abschnitte 7.5 und 7.6 dokumentieren die Ergebnisse der Tests für die Bilddaten der CEDAR und der NIST SD3 Datenbanken. Diese Messungen bilden eine solide Grundlage, mit der off-line Erkennungssysteme für handgeschriebene Zahlen verglichen werden können.

Um eine bessere Vergleichbarkeit der Resultate zwischen den einzelnen Abschnitten zu ermöglichen, beziehen sich die hier dokumentierten Experimente (bis auf den letzten Abschnitt) ausschliesslich auf Bilddaten der NIST SD3 Datenbank. Wenn die verwendeten Bilddaten nicht näher spezifiziert sind, handelt es sich um die 1275 zwei- bis sechsstelligen Zahlen der Formulare 'f0000' bis 'f0050' (25 Zahlen pro Formular).

7.1 Optimierung der Parameter

Wie im Kapitel 6 dargelegt, wird das Verhalten des Zahlenerkenners durch eine Reihe von Parametern beeinflusst. Da die optimalen[1] Werte dieser Parameter von den zu erkennenden Bilddaten, der eingesetzten Ziffernerkennung und der Verknüpfung der

[1] Die Optimalität ist hier im Sinne einer möglichst hohen Erkennungsrate zu verstehen.

einzelnen Module abhängen, müssen diese experimentell durch ein systematisches Vorgehen bestimmt werden.

Um für alle Parameter (siehe Tabelle 7.1) vernünftige Startwerte zu finden, wurden zu Beginn der Optimierungsphase einige kleine Experiemente durchgeführt. Dabei hat sich gezeigt, dass eine weitere Optimierung von einigen Parametern nicht mehr notwendig ist.

Im Modul 'Vorsegmentierung' müssen die Parameter $T_{Vorsegment1}$ und $T_{Vorsegment}$ so eingestellt sein, dass es nicht zur isolierten Interpretation von Bruchstücken kommen kann. Im Modul 'Erkennung Teilbild' dürfen die Parameter $T_{Teilerk1}$ und $T_{Teilerk2}$ die Generierung der Ziffernbilder nicht so stark einschränken, dass korrekte Ziffernbilder unterdrückt werden. Anhand der oben angesprochenen Experimente konnten für diese Parameter bereits definitive Werte gefunden werden, die deshalb in Tabelle 7.1 als "statisch" bezeichnet sind. Da zur Erkennung der NIST SD3 Bilddaten das Konzept des 'Dummy-Segmentes' nicht eingesetzt wurde, sind die Werte der beiden letzten Parameter W_{Dummy1} und W_{Dummy2} in Klammern gesetzt.

Auf die beiden Schwellwertfunktionen $T_{Entscheidung1}(t)$ und $T_{Entscheidung2}(t)$ des Moduls 'Kombination & Entscheidung' wird hier nicht weiter eingegangen, da ihre Optimierung im Abschnitt 7.3 besprochen wird.

Modul	Parameter	Wert	Optimierung
Vorsegmentierung	$T_{Vorsegment_1}$	0.50	statisch
	$T_{Vorsegment_2}$	3.30	statisch
	$T_{Vorsegment_3}$	0.20	zu optimieren
Detektion Ziffern	$T_{Detection}$	0.05	zu optimieren
Erkennung Teilbild	$T_{Teilerk1}$	0.30	statisch
	$T_{Teilerk2}$	4.00	statisch
	W_{Knoten}	0.50	zu optimieren
	W_{Dummy1}	(30.0)	(zu optimieren)
	W_{Dummy2}	(2.0)	(zu optimieren)

Tabelle 7.1: Parameter des Zahlenerkenners (mit Startwerten)

Um eine gute Konfiguration der Parameter zu finden, müssen somit in der Optimierungsphase nur $T_{Vorsegment3}$ (TV), $T_{Detektion}$ (TD) und W_{Knoten} (WK) betrachtet werden. Durch die gegebne Verknüpfung der Module können diese drei Parameter jedoch nicht unabhängig voneinander optimiert werden. Ausgehend von den in Tabelle 7.1 angegebenen Startwerten wurde ein (zu optimierender) Parameter nach dem anderen um einen kleinen Betrag verändert, und der Effekt der entsprechenden Veränderung anschliessend durch die Messung der Erkennungsrate überprüft.

Tabelle 7.2 gibt diese Experimente zur Optimierung der Parameter TV, TD und WK

Experiment	TV	TD	WK	Beispiele	Korrekt	Unter/Über	%
1	0.20	0.050	0.50	1275	1046	15/112	82.04
2	0.10*	0.050	0.50	1275	1042	17/113	81.73
3	0.30*	0.050	0.50	1275	1042	15/116	81.73
4	0.20	0.020*	0.50	1275	1087	21/39	85.26
5	0.20	0.010*	0.50	1275	1078	56/19	84.55
6	0.20	0.020	0.70*	1275	1090	18/37	85.49
7	0.20	0.020	0.80*	1275	1095	18/37	85.88
8	0.10*	0.020	0.80	1275	1099	19/33	86.19
9	0.05*	0.020	0.80	1275	1100	19/32	86.27
10	0.05	0.015*	0.80	1275	1094	32/23	85.80
11	0.05	0.018*	0.80	1275	1098	24/30	86.12
12	0.05	0.020	0.90*	1275	1104	19/29	86.59
13	0.05	0.020	0.95*	1275	1104	19/33	86.59

Tabelle 7.2: Optimierung der Parameter des Zahlenerkenners

wieder (die veränderten Werte dieser Parameter sind durch einen Stern gekennzeichnet). Die Spalte 'Beispiele' enthält die Anzahl der getesteten Beispiele, die Spalte 'Korrekt' die Anzahl der korrekt erkannten Beispiele und die Spalte 'Unter/Über' die Anzahl der unter- beziehungsweise übersegmentierten Beispiele. Die Spalte '%' beinhaltet die gemessenen Erkennungsraten des Zahlenerkenners.

Der optimierte Zahlenerkenner wurde anschliessend mit der Parameterkonfiguration des Experimentes 12 (siehe Tabelle 7.2) anhand der Beispiele der Testdaten geprüft. Die Testdaten umfassen dabei die zwei- bis sechsstelligen Zahlen der NIST SD3 Formulare 'f1800' bis 'f1899' und 'f2000' bis 'f2099' ohne das Formular 'f1823', da hier die Extraktion der Zahlenfelder scheiterte. Die Tabelle 7.3 fasst die Resultate für diese Testdaten zusammen.

Experiment	TV	TD	WK	Beispiele	Korrekt	Unter/Über	%
A	0.05	0.020	0.90	4875	4219	109/94	86.54
B	0.05	0.020	0.90	4875	4403	96/93	90.32

Tabelle 7.3: Test des Zahlenerkenners

Die Erkennungsrate im Experiment 'A' (Testdaten) stimmt sehr gut mit der Erkennungsrate des Experiementes 12 (Trainingsdaten) überein, da hier derselbe Ziffernerkenner zum Einsatz kam (mit Ziffern der CEDAR Datenbank trainiert). Im Laufe dieser Diplomarbeit wurde später auch der mit NIST SD3 Bilddaten trainierte Ziffernerkenner verfügbar, wodurch die Erkennungsrate des Zahlenerkenners im Experiment 'B' signifikant gestiegen ist.

7.2 Fehleranalyse

In diesem Experiment sollen die verschiedenen Fehlerquellen lokalisiert und quantifiziert werden. Dazu wurden sieben verschiedene Fehlerarten definiert, um die Zuverlässigkeit der einzelnen Module des Zahlenerkenners untersuchen zu können (siehe Tabelle 7.4). Für diese Untersuchung ist die Konfiguration des Zahlenerkenners des Experimentes 'B' (Tabelle 7.3) und die 1275 Beispiele der Experimente 1-13 (Tabelle 7.2) verwendet worden.

Art	Bezeichnung	verantwortliches Modul
F1	Wahrheitswert-Fehler	-
F2	Extraktions-Fehler	-
F3	Gruppierungs-Fehler	'Vorsegmentierung'
F4	Detektions-Fehler	'Detektion Ziffern'
F5	Ziffern-Fehler	'Ziffernerkenner'
F6	Zifferngruppen-Fehler	'Erkennung Teilbild'
F7	Schwierige Zahlen	-

Tabelle 7.4: Definition der untersuchten Fehlerarten

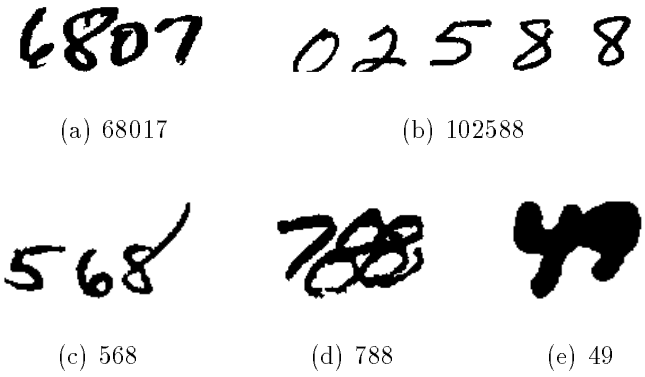
Die Fehlerart 'F1' steht für fehlerhafte Beispiele, bei denen der Bildinhalt nicht mit dem Wahrheitswert übereinstimmt. Mit 'F2' werden Beispiele bezeichnet, welche durch die Extraktion der Zahl aus dem Formular Ziffern oder Teile von Ziffern verloren haben (siehe Abbildung 7.1 (a) und (b)). 'F3' bezeichnet Fehler, die durch ein Fehlverhalten des Moduls 'Vorsegmentierung' entstanden sind. Wenn eine Zifferngruppe im Modul 'Detektion Ziffern' fälschlicherweise als isolierte Ziffer klassifiziert wurde stellt dies ein 'F4'-Fehler dar[2]. Korrekt segmentierte Zahlen werden dem integrierten Ziffernerkenner als 'F5'-Fehler angelastet und Fehler, die im Modul 'Erkennung Teilbild' entstanden sind, werden mit 'F6' bezeichnet[3]. Die letzte Fehlerart 'F7' betrifft Beispiele, die für den vorgestellten Zahlenerkenner grundsätzlich nicht erkennbar sind (siehe Abbildung 7.1 (c), (d) und (e)).

Tabelle 7.5 zeigt die Aufteilung der 130 entstandenen Erkennungsfehler auf die oben definierten Fehlerarten. Die Erkennungsfehler wurden zusätzlich in die zwei folgenden Kategorien unterteilt:

- Unter- und übersegmentierte Zahlen (Spalte 'Unter/Über'): Dies betrifft Beispiele, deren Wahrheitswert mehr beziehungsweise weniger Ziffern als das Erkennungsresultat enthält.

[2]Das Klassifizieren einer isolierten Ziffer als Zifferngruppe stellt hier keinen Fehler dar, da das Modul 'Erkennung Teilbild' prinzipiell in der Lage ist, auch diese Fälle korrekt zu erkennen.

[3]In dieser Kategorie sind sowohl Fehler der Segmentierung als auch Fehler des Ziffernerkenners enthalten.



(a) 68017 (b) 102588

(c) 568 (d) 788 (e) 49

Abbildung 7.1: Beispiele zu verschiedenen Fehlerartenn

- Andere Erkennungsfehler (Spalte 'Andere'): Der Wahrheitswert dieser Beispiele enthält die gleiche Anzahl Ziffern wie das Erkennungsresultat.

Art	Unter/Über	Andere	Summe
F1	2	3	5
F2	2	0	2
F3	0	1	1
F4	7	0	7
F5	0	47	47
F6	17	10	27
F7	10	31	41
alle	38	92	130

Tabelle 7.5: Verteilung der Fehler auf die Fehlerarten

Dieses Experiment zeigt, dass die meisten Erkennungsfehler (47) durch Fehlinterpretation von Ziffern entstehen. Da die Ziffernerkennung ausserdem für Erkennungsfehler der Art 'F6' und 'F7' mitverantwortlich ist, muss der Anteil der Erkennungsfehler, der auf die Ziffernerkennung zurückgeführt werden kann, auf über 50% geschätzt werden.

7.3 Rückweisungsregel

Durch die hier vorgestellten Experimente sollen die wichtigsten Schritte der Entwicklung der eingesetzten Rückweisungsregel nachvollzogen werden. Alle aufgeführten Messungen wurden mit der im letzten Abschnitt beschriebenen Konfiguration des Zahlenerkenners durchgeführt.

In der ersten Phase wurde nach einem beispielabhängigen Wert gesucht, durch den die Korrektheit der berechneten Interpretation abgeschätzt werden kann. Für jede Ziffer einer Interpretation sind prinzipiell zwei Werte aus dem Erkennungsprozess verfügbar: Die Wertung aus dem entsprechenden Resultat R_{Ziffer} und das Mass Q_{Ziffer} (siehe Abschnitt 6.5). Falls diese Werte tatsächlich ein Mass für die Sicherheit der Erkennung der einzelnen Ziffern darstellen, folgt daraus, dass die Sicherheit der Erkennung der ganzen Zahl nicht höher sein kann, als das Minimum der Sicherheiten aller enthaltenen Ziffern.

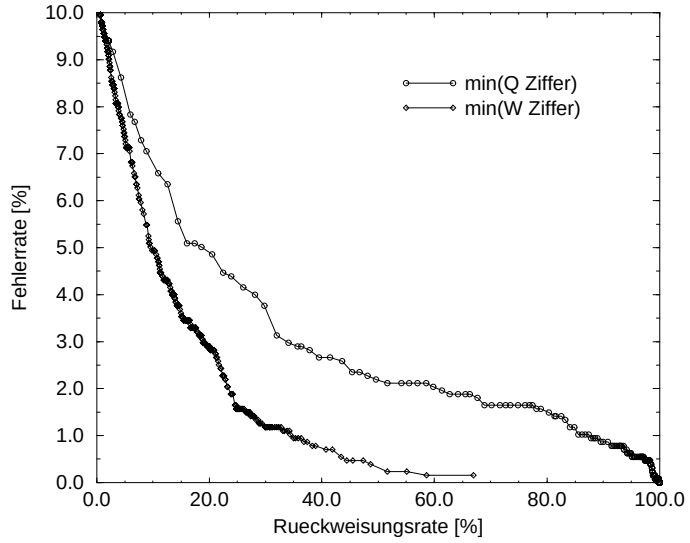


Abbildung 7.2: Fehlerraten und Rückweisungsraten für verschiedene Kriterien

Die erste implementierte Rückweisungsregel testete $min(Wertung(R_{Ziffer}(i))) = Q_{Zahl}$ (siehe Abschnitt 6.7) gegen einen fixen Schwellwert $T \in [0,1]$. Ein Beispiel wurde also akzeptiert, wenn der berechnete Wert Q_{Zahl} höher oder gleich dem Schwellwert T lag. Andernfalls wurde das Beispiel zurückgewiesen. In einem zweiten Versuch wurde der Wert Q_{Zahl} durch $min(Q_{Ziffer}(i))$ ersetzt. Um diese beiden Regeln miteinander vergleichen zu können, wurden für verschiedene Schwellwerte

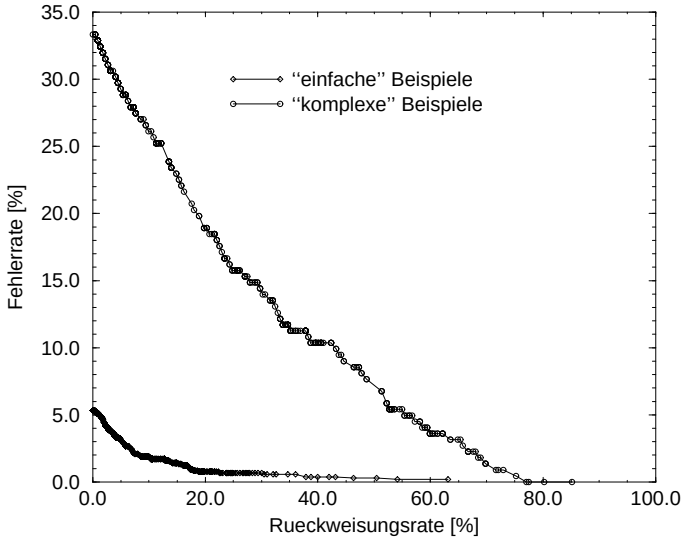


Abbildung 7.3: Fehlerraten und Rückweisungsraten für "einfache" und "komplexe" Beispiele

von 0 bis 1 die zugehörigen Fehlerraten und Rückweisungsraten gemessen und graphisch dargestellt. Das Ergebnis dieser Experimente ist in Abbildung 7.2 durch die beiden Kurven 'min(W Ziffer)' = $min(Wertung(R_{Ziffer}(i)))$ und 'min(Q Ziffer)' = $min(Q_{Ziffer}(i))$ dargestellt. Die Analyse der beiden Kurven legte die Verwendung des Wertes min(W Ziffer)= Q_{Zahl} als Grundlage für Rückweisungsentscheide nahe.

Durch die Unterteilung der Bilddaten in "einfache" und "komplexe" Beispiele (siehe Abschnitt 6.7), fiel auf, dass die Erkennungsrate für "einfache" Beispiele viel höher war als bei der Interpretation von "komplexen" Beispielen. Abbildung 7.3 veranschaulicht den massiven Unterschied der gemessenen Erkennungsraten.

Diese Beobachtungen haben dazu geführt, dass die Rückweisungsregel modifiziert wurde. Die neue Regel sollte in erster Linie "komplexe" Beispiele zurückweisen. Dazu wurden zwei unabhängige Schwellwerte für "komplexe" und "einfache" Bilder eingeführt, die in Abschnitt 6.7 mit $T_{Entscheidung1}$ und $T_{Entscheidung2}$ bezeichnet sind.

Die Optimierung des Parameter P erfolgte zuerst getrennt für die vorgegebenen Fehlerraten von 2%, 1% und 0.5%. Der "optimale" Wert von 0.85 wurde anschliessend durch die Mittelung der optimalen Werte für die vorgegebenen Fehlerra-

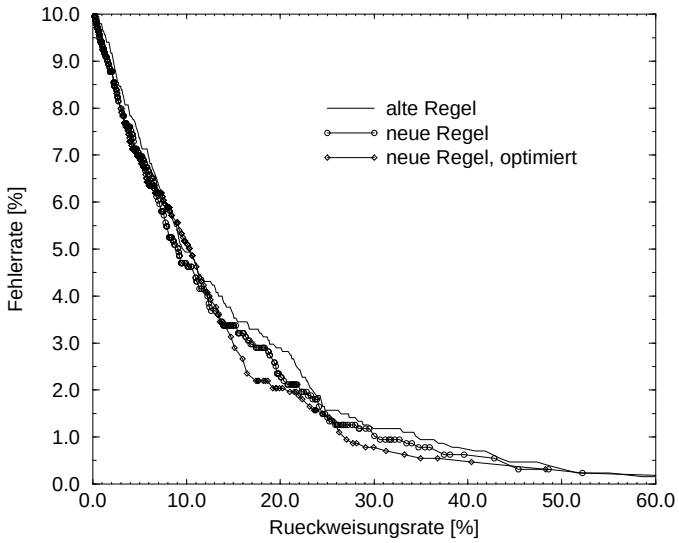


Abbildung 7.4: Fehlerraten und Rückweisungsraten für die alte und die neue Rückweisungsregel

Regel	2.0%	1.0%	0.5%
alt	23.4	34.4	44.0
neu, optimiert	19.2	26.4	38.0

Tabelle 7.6: Vergleich der Rückweisungsraten für verschiedene Fehlerraten

ten bestimmt[4]. Wie in Abbildung 6.21 dargestellt, wird die Schwellwertfunktion $T_{Entscheidung2}(t)$ durch den Punkt P auf der t-Achse festgelegt. Abbildung 7.4 vergleicht die alte Regel ("einfache" und "komplexe" Beispiele werden gleich behandelt) mit der neuen Rückweisungsregel ($P = 0.5$) und der optimierten neuen Rückweisungsregel ($P = 0.85$). Tabelle 7.6 dokumentiert den Einfluss der neuen, optimierten Entscheidungsregel auf die Rückweisungsrate für die vorgegebenen Fehlerraten 2%, 1% und 0.5%.

[4]Dieses Optimierungsverfahren wurde gewählt, da zu diesem Zeitpunkt die Definition des Masses Q_{Regel} (siehe Abschnitt 6.7) noch nicht entworfen war.

7.4 Diverse Experimente

Dieser Abschnitt fasst die Resultate verschiedener Experimente zusammen, die durchgeführt wurden, um die Vorverarbeitung[5] der Beispiele und die Zuverlässigkeit der Ziffernerkennung zu verbessern.

- Eine Normierung der Schriftneigung, wie sie in Abschnitt 5.3 beschrieben wurde, hat keine Verbesserungen der Erkennungsrate des Zahlenerkenners gebracht.
- Durch die Normierung der Schriftorientierung (Skew Normalisation) wurde nur für die Bilddaten der CEDAR Datenbank eine Verbesserung der Erkennungsrate erzielt.
- Das Konzept des 'Dummy-Segmentes' ist nur für Bilddaten der CEDAR Datenbank geeignet, da diese im Gegensatz zu NIST SD3 Bilddaten für den Zahlenerkenner viele uninterpretierbare Bildteile enthalten[6].
- Durch den Einsatz eines besseren Ziffernerkenners, der auf der 'Perturbation-Methode' basiert [HB94], wurde für die Rückweisungsrate 0 praktisch keine Verbesserung erzielt. Für Rückweisungsraten > 0 waren die Erkennungsraten des Zahlenerkenners sogar deutlich tiefer als beim Einsatz des ursprünglichen Ziffernerkenners.

[5] Die entsprechenden Schritte erfolgen im Modul 'Vorsegmentierung' vor dem Schritt 'ZK-Extraktion & Filterung'.

[6] Um für NIST SD3 Bilddaten optimale Erkennungsraten zu erreichen, mussten die Parameter W_{Dummy1} und W_{Dummy2} so hoch angesetzt werden, dass das Konzepte des Dummy-Segmentes gar nicht mehr zur Anwendung gekommen ist.

7.5 Resultate für die CEDAR Datenbank

Dieser Abschnitt präsentiert die Resultate der Tests des Zahlenerkenners auf den CEDAR Bilddaten. Der eingesetzte Zahlenerkenner verwendetete eine Parameterkonfiguration die dem Abschnitt 7.1 entsprechend, auf den ersten 500 Beispielen des Verzeichnisses *train/zipcodes/br* der CEDAR Datenbank optimiert wurde. Der Zahlenerkenner führte eine Normierung der Schriftorientierung vor der Erkennung durch und liess Interpretationen mit 'Dummy-Segmenten' zu.

Der vom Zahlenerkenner verwendete Ziffernerkenner verwendetet die 'Perturbation-Methode' nicht und ist mit den 18468 Ziffern des Verzeichnisses *train/bindigis* trainiert worden. Für die Rückweisungsrate 0 erreichte der Ziffernerkenner eine Erkennungsrate von 98.69% (getestet auf den 2213 Ziffern des Verzeichnisses *test/bindigis/goodbs*).

Der Test des Zahlenerkenners mit den CEDAR Bilddaten wurde in zwei Experimente aufgeteilt. Im ersten Experiment wurden die 495 Beispiele des Verzeichnisses *test/binzips/bs* verwendet und im zweiten Experiment alle 435 Beispiele des Verzeichnisses *test/zipcodes*[7]. Für das erste Experiment fasst Tabelle 7.7 die gemesse-

Länge	Beispiele	Null	2%	1%	0.5%
5,9	495	83.6	60.0	51.5	48.0
5	436	84.9	64.0	54.5	50.0

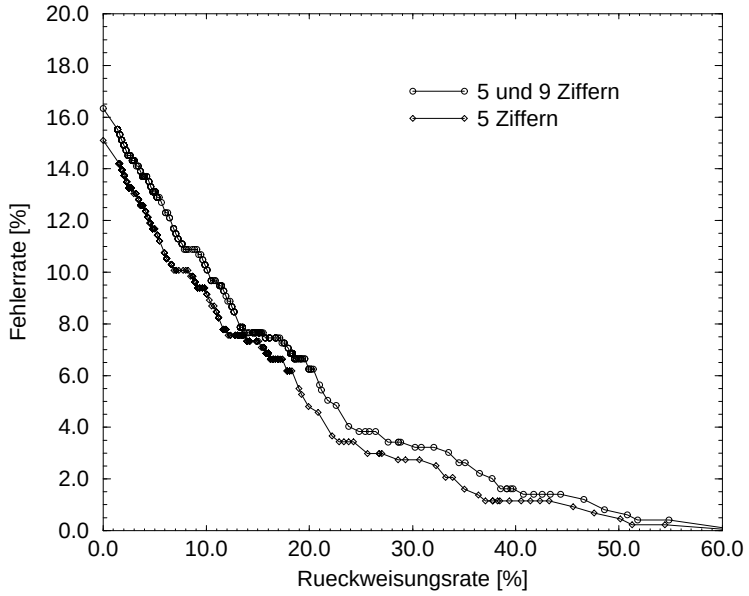
Tabelle 7.7: Erkennungsraten für CEDAR *test/binzip* Bilddaten

nen Erkennungsraten zusammen. In der Spalte 'Null' sind die Erkennungsraten für die Rückweisungsrate 0 angegeben und die Spalten '2%', '1%' und '0.5%' enthalten die Erkennungsraten für die entsprechend vorgegebenen Fehlerraten. Die Spalte 'Länge' enthält die Anzahl der Ziffer der untersuchten Beispiele. Abbildung 7.5 (a) dokumentiert das Verhalten des Zahlenerkenners für alle Rückweisungsraten des ersten Experimentes. Tabelle 7.8 und Abbildung 7.5 (b) enthalten die entsprechenden Ergebnisse für die *test/zipcodes* Bilddaten.

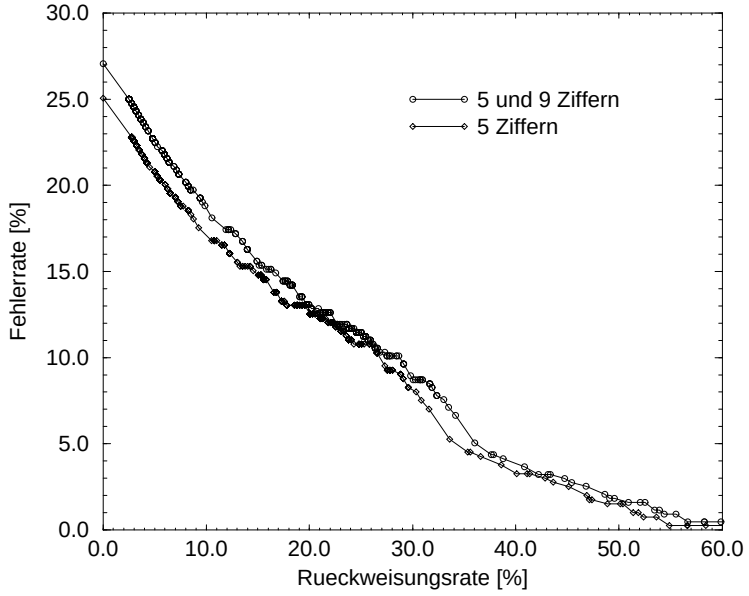
Länge	Beispiele	Null	2%	1%	0.5%
5,9	435	72.9	49.5	44.5	43.0
5	398	74.9	51.0	47.5	45.0

Tabelle 7.8: Erkennungsraten für CEDAR *test/zipcodes* Bilddaten

[7] Alle Bilder dieses Verzeichnisses weisen 256 Graustufen auf. Vor ihrer Erkennung wurden diese Bilder mit der Methode von N. Otsu binärisiert [Ots79].



(a)



(b)

Abbildung 7.5: Fehlerraten und Rückweisungsraten für CEDAR *test/binzip* Bilddaten(a) und *test/zipcodes* Bilddaten (b).

7.6 Resultate für die NIST SD3 Datenbank

Dieser Abschnitt präsentiert die Resultate der Tests des Zahlenerkenners auf den NIST SD3 Bilddaten. Der eingesetzte Zahlenerkenner verwendetet die Parameterkonfiguration des Experimentes 12 (Tabelle 7.2), führt keine Normierung der Schriftneigung oder der Schriftorierntierung durch und lässt die Verwendung von 'Dummy-Segmenten' nicht zu.

Der vom Zahlenerkenner verwendete Ziffernerkenner verwendetet die 'Perturbation-Methode' nicht und ist mit den letzten 173124 Ziffern des Verzeichnisses *SD3/data* trainiert worden. Für die Rückweisungsrate von 0 erreichte der Ziffernerkenner eine Erkennungsrate von 99.45%.

Für dieses Experiment wurden die Bilddaten verwendet, die bereits in Abschnitt 7.1 bei den Experimenten 'A' und 'B' zum Einsatz kamen. 13 Beispiele der Fehlerart 'F1'[8] und 35 Beispiele der Fehlerart 'F2'[9] (siehe Abschnitt 7.2) wurden jedoch aus den Testdaten entfernt, um zuverlässigere Schätzungen der Fehler- und Rückweisungsraten zu erzielen. Tabelle 7.9 fasst die gemessenen Erkennungsraten zu-

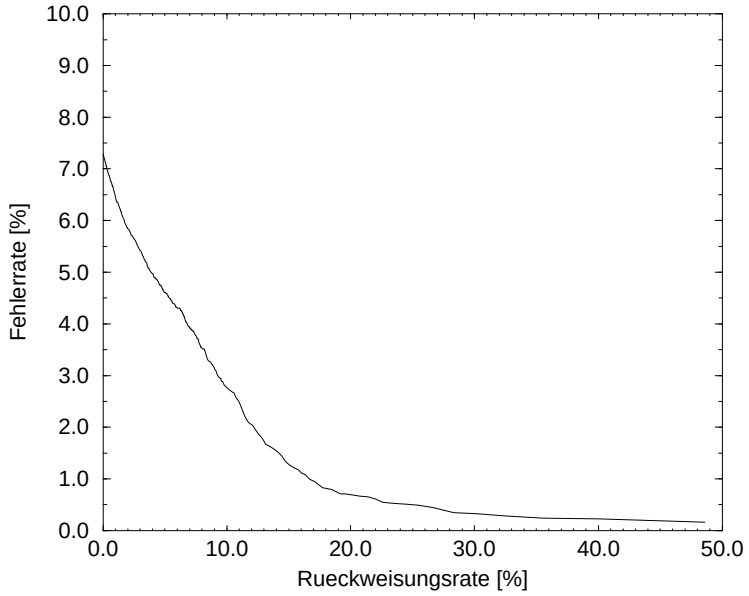
Länge	Beispiele	Null	2%	1%	0.5%
2	981	96.2	94.5	93.5	91.5
3	986	92.7	86.0	79.5	70.5
4	988	93.2	86.0	81.0	70.0
5	988	91.1	81.0	77.5	70.5
6	982	90.3	80.5	75.5	66.5
2-6	4925	92.7	86.0	82.0	74.0

Tabelle 7.9: Erkennungsraten für NIST Bilddaten

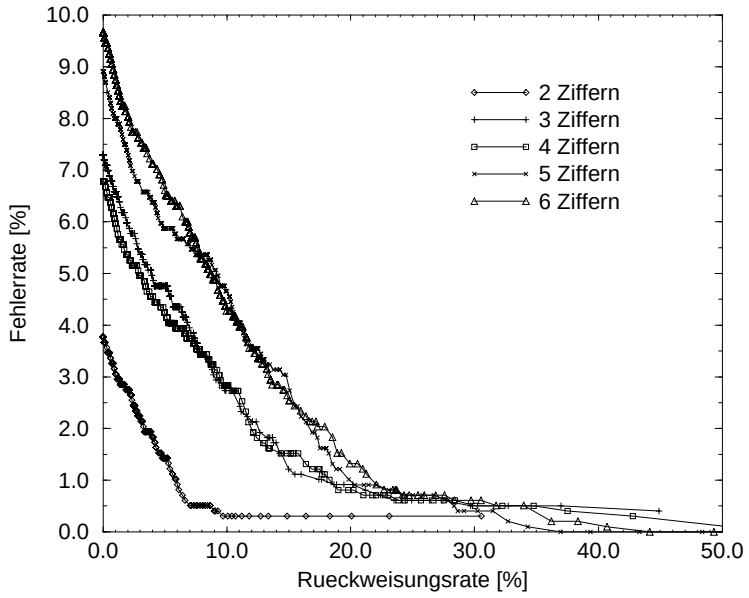
sammen. In der Spalte 'Null' sind die Erkennungsraten für die Rückweisungsrate 0 angegeben und die Spalten '2%', '1%' und '0.5%' enthalten die Erkennungsraten für die entsprechend vorgegebenen Fehlerraten. Die Spalte 'Länge' spezifiziert den untersuchten Feldtyp durch die Anzahl der enthaltenen Ziffern. Abbildung 7.6 (a) dokumentiert das (über alle Längen gemittelte) Verhalten des Zahlenerkenners für alle Rückweisungsraten. Abbildung 7.6 (b) schlüsselt die Fehlerraten für die unterschiedlichen Feldtypen weiter auf.

[8] `f1808[4], f1824[18], f1828[9], f1861[18], f1890[7], f2001[14], f2003[19], f2030[6], f2035[19], f2056[8], f2070[15], f2076[8], f2080[19]`.

[9] `f1801[23], f1830[15], f1874[19], f1874[20], f1876[17], f1892[16], f1896[7], f1896[24], f1898[19], f2024[17], f2025[16], f2027[9], f2027[10], f2027[12], f2027[15], f2027[16], f2027[20], f2027[21], f2027[22], f2027[25], f2027[26], f2027[27], f2048[19], f2055[12], f2055[24], f2056[14], f2056[22], f2067[15], f2067[24], f2067[25], f2067[8], f2089[14], f2093[5], f2095[22], f2096[16]`.

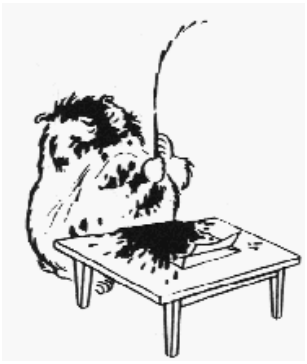


(a)



(b)

Abbildung 7.6: Fehlerraten und Rückweisungsraten für NIST Bilddaten, für alle Feldtypen gemittelt (a) und pro Feldtyp (b).



Kapitel 8

Diskussion und Vergleich mit publizierten Methoden

Dieses Kapitel diskutiert wichtige Ergebnisse der durchgeführten Experimente und vergleicht diese mit den publizierten Resultaten anderer Arbeiten.

In Abschnitt 8.1 werden zuerst die erreichten Resultate mit der Aufgabenstellung dieser Diplomarbeit verglichen, die Stärken und Schwächen des vorgestellten Zahlenerkenners aufgezeigt und interessant erscheinende, weiterführende Arbeiten vorgeschlagen. Im Abschnitt 8.2 werden die gemessenen Erkennungsraten auf den CEDAR und NIST SD3 Bilddaten mit den publizierten Resultaten anderer Autoren verglichen.

8.1 Diskussion

Dieser Abschnitt ist in drei Unterabschnitte aufgeteilt. Im ersten Unterabschnitt werden die in Abschnitt 1.1 vorgestellten Ziele mit den erreichten Resultaten verglichen. Der zweite Unterabschnitt geht auf die beobachteten Stärken und Schwächen des Zahlenerkenners. Der letzte Unterabschnitt enthält einige Anregungen für weiterführende Arbeiten.

8.1.1 Zielerreichung

Das erste Ziel dieser Arbeit bestand darin, das von Dr. Thien Ha Minh und Dieter Niggeler vorgeschlagene Erkennungssystem zu verbessern und zu erweitern. In der

folgenden Aufstellung wird auf die in [Nig94] vorgeschlagenen Verbesserungen und Erweiterungen eingegangen.

- Die in [Nig94] festgestellte Hauptfehlerquelle, der 'Ziffernerkenner', stand als Rahmenbedingung dieser Diplomarbeit fest. Seine Leistungsfähigkeit ist nicht verbessert worden. Diesbezügliche Experiemente (siehe Abschnitt 7.2) quantifizierten und bestätigten die Feststellung von Dieter Niggeler.

  Der Einsatz eines Ziffernerkenners, der auf der 'Perturbation-Methode' basierte war erfolglos. Dieser Misserfolg kann möglicherweise dadurch begründet werden, dass der eingesetzte Ziffernerkenner die Möglichkeit hatte, für jede Ziffer einer Zahl ein anderes 'Perturbation'-Modell auszuwählen, was im Modul 'Erkennung-Teilbild' eine zu optimistische Bewertung vieler generierter Ziffernbilder zur Folge hatte.

- Die Normierung der Schriftneigung vor der Erkennung hat ebenfalls zu keiner signifikanten Verbesserung der Erkennungsresultate geführt. In gewissen Fällen wurden einzelne Ziffern, die sich vor der Normierung nicht berührten, zu einer Zusammenhangskomponente verschmolzen, die anschliessend durch das fehleranfälligere Modul 'Erkennung Teilbild' erkannt werden mussten. Es scheint, dass dieser Effekt die für andere Beispiele erreichten Verbesserungen zunichte gemacht hat.

- Der vorgeschlagene Pertrubation-Ansatz für das Modul 'Erkennung Teilbild' ist nicht untersucht worden.

- Die vorgeschlagene Erweiterung der Systemarchitektur durch einen Paarerkenner kann als gescheitert betrachtet werden (siehe Abschnitt 5.6).

Die tatsächlich erzielten Verbesserungen lassen sich leider nicht objektiv angeben, da in [Nig94] nicht zwischen Trainings- und Testdaten unterschieden worden ist. Die in dieser Diplomarbeit eingebauten Verbesserungen und Erweiterungen sind in der folgenden Aufstellung zusammengefasst.

- Die wichtigste Erweiterung betrifft das Modul 'Kombination & Entscheidung'. Durch einen Rückweisungsmechanismus der in dieser Diplomarbeit entwickelt, optimiert und getestet wurde, kann der Zahlenerkenner auf eine vorgegebenen Fehlerrate oder Rückweisungsrate eingestellt werden.

  Diese Erweiterung erlaubt den Vergleich der Erkennungsleistung des Zahlenerkenners mit publizierten Ergebnissen für den ganzen Rückweisungsbereich von 0 bis 100%.

- Das neuentwickelte Verfahren zur Extraktion der Segmente (siehe Abschnitt 6.6) reduziert die ursprünglich quadratische Zeitkomplexität (zur Anzahl der Bildpunkte) für das "Reverse-Thinning" auf lineare Zeitkomplexität.

Das zweite Ziel bestand darin, den Zahlenerkenner mit publizierten Forschungsarbeiten zu vergleichen. Dazu waren umfangreiche Tests auf den vorgestellten Datenbanken erforderlich. Der Vergleich der erzielten Resultate mit publizierten Ergebnissen wird im Abschnitt 8.2 gezogen.

8.1.2 Stärken und Schwächen des Zahlenerkenners

Die Stärken des hier vorgestellten Zahlenerkenners beruhen einerseits auf der Verwendung eines leistungsfähigen Ziffernerkenners und andererseits auf der Kombination einer segmentierungsbasierten und einer segmentierungsfreien Erkennungsmethode. Die implementierte Architektur des Zahlenerkenners stellt ein flexibles Framework zur Verfügung, das keine spezielle Technik zur Erkennung von isolierten Ziffern oder Zifferngruppen voraussetzt. Die Tauglichkeit dieses Konzeptes wird im Abschnitt 8.2 im Vergleich mit publizierten Erkennungssystemen nachgewiesen.

Da die verwendeten (realitätsnahen) Bilddaten in der Regel leserlich und klar segmentiert sind, können viele ("einfache") Beispiele ohne Einsatz des Modules 'Erkennung Teilbild' mit hoher Zuverlässigkeit erkannt werden.

In der Tabelle 8.1 sind die Anteile der "einfachen" und "komplexen" Beispiele für die in Abschnitt 7.6 definierten Testdaten der NIST SD3 Datenbank zusammengefasst. Die "komplexen" Beispiele wurden in Spalte 'Art' weiter in die Kategorien 'k1', 'k2' und 'k3' aufgeteilt (die Bedeutung der restlichen Spalten ist dieselbe wie bei Tabelle 7.2). Die Art eines "komplexen" Beispiels wird durch das längste Resultat R_{Teil} festgelegt, das vom Modul 'Erkennung Teilbild' ausgegeben wird. Eine Zahl, die isolierte Ziffern und ein Ziffernpaar enthält, wäre somit von der Art 'k2' (falls das Paar vom Zahlenerkenner als solches erkannt wird).

Art	Beispiele	Korrekt	Über/Unter	%
"einfach"	4096	3975	7/12	97.05
"komplex"	829	591	65/28	71.29
k1	252	167	22/14	66.27
k2	506	391	27/10	77.27
k3	67	32	16/3	47.76
Summe	4925	4566	72/40	92.71

Tabelle 8.1: Verteilung der Beispiele für die NIST Testdaten

Tabelle 8.1 zeigt, dass 83% der NIST SD3 Bilddaten aus "einfachen" Beispielen besteht, die mit einer sehr hohen Sicherheit erkannt werden können (Erkennungsrate 97.05%)[1].

[1]Für die CEDAR Testdaten entspricht der Anteil der "einfachen" Beispiele 68% (Verzeichnis

Die oben angeführte Stärke des Kombinations-Konzeptes führt in der Praxis zu einer komplexen Implementierung, die aus einer Vielzahl von Komponenten und Parametern besteht, die sich gegenseitig beeinflussen. Für die Beherrschbarkeit dieses Systemes sind dies keine günstigen Voraussetzungen.

Eine andere Schwäche des Zahlenerkenners stellen die (im Vergleich zu [LL95]) "tiefen" Erkennungsraten für Zahlen dar, die nur aus zwei oder drei Ziffern bestehen.

Eine weitere Schwäche des Erkennungssystems ist aus Tabelle 8.1 ersichtlich. Die Erkennungsrate für "komplexe" Beispiele mit Zifferngruppen, die mehr als zwei Ziffern enthalten, ist mit 47% sehr tief.

8.1.3 Weiterführende Arbeiten

An dieser Stelle werden einige Vorschläge für weiterführende Arbeiten gemacht, die für die bessere Erkennung von handgeschriebenen Zahlen interessant sein könnten. Sie sind nach dem geschätzten Aufwand (aufsteigend) sortiert.

- Die in Abschnitt 6.7 angegebene Schwellwertfunktion $T_{Entscheidung2}(t)$ ist sehr einfach. Möglicherweise kann die vorgeschlagene Rückweisungsregel durch den Einsatz einer geeigneteren Funktion weiter verbessert werden.

  Tabelle 8.1 legt ausserdem den Schluss nahe, dass die Rückweisungsregel nicht nur zwischen "einfachen" und "komplexen" Beispielen unterscheiden sollte, sondern auch die für die Beispielarten 'k1', 'k2' ... verschiedenen Erkennungsraten berücksichtigen könnte.

  Um diese Problematik weiter zu verfolgen, wäre es ausserdem interessant, wenn sich die Optimierung der Rückweisungsregeln auf das in Abschnitt 6.7 vorgeschlagene Mass Q_{Regel} beziehen würde.

- Da die meisten Erkennungsfehler auf den Ziffernerkenner zurückgeführt werden können, müsste sich die Erkennungsrate des Zahlenerkenners durch einen zuverlässigeren Ziffernerkenner signifikant steigern lassen (beispielsweise durch die Kombination von mehr als zwei Erkennungsmethoden zur Ziffernerkennung, siehe [SNL+92]).

- Die Konstruktion einer Funktion zur zuverlässigen Schätzung der Anzahl in einer Zifferngruppe enthaltenen Ziffern könnte helfen, die relativ hohen Fehlerraten bei der Segmentierung von "komplexen" Beispielen zu verbessern (entsprechende Gewichtung der Kosten eines Lösungspfades).

test/binzips) beziehungsweise 59% (Verzeichnis *test/zipcodes*).

- Falls die oben genannte Schätzung realisiert wäre, könnte damit ein segmentierungsfreies Erkennungsverfahren implementiert werden, das auf einem Clustering extrahierter Segmente basieren würde. Ein solches Verfahren wäre flexibler und einfacher zu realisieren, als das hier beschriebene Modul 'Erkennung Teilbild'.

- Durch eine Kombination des vorgestellten Systems mit einem weiteren Zahlenerkenner, der auf einer kontinuierlichen Segmentierungsmethode basiert, könnten wahrscheinlich noch höhere Erkennungsraten erreicht werden.

8.2 Vergleich mit publizierten Methoden

Der saubere Vergleich der in dieser Diplomarbeit erreichten Resultate mit publizierten Resultaten anderen Arbeiten ist schwierig, da bisher von keinem Erkennungssystem Resultate für beide Datenbanken (CEDAR und die NIST SD3) publiziert worden sind. Ausserdem ist für viele Messungen auf den NIST Bilddaten nicht bekannt, welche Beispiele für die Tests verwendet wurden.

Dieser Abschnitt ist in zwei Unterabschnitte gegliedert. Im ersten Unterabschnitt wird das hier vorgestellte System zur Erkennung von handgeschriebenen Zahlen mit Arbeiten von Autoren verglichen, die für ihre Tests CEDAR Bilddaten verwendet haben. Im zweiten Unterabschnitt werden die Resultate für die NIST SD3 Bilddaten den entsprechenden, publizierten Arbeiten gegenübergestellt.

8.2.1 Vergleich für CEDAR Bilddaten

Die erreichten Erkennungsraten auf den CEDAR Bilddaten sollen hier nur mit Resultaten von Arbeiten verglichen werden, die unter vergleichbaren Testbedingungen entstanden sind.

Das von F. Kimura und M. Shridar in [KS92] vorgeschlagene Erkennungssystem basiert auf dem segmentierungsfreien Ansatz, der Zifferngruppen durch gerade Linien in isolierte Ziffern unterteilt. Als Testdaten wurden die Beispiele des Verzeichnisses *test/binzips* verwendet, wobei 16 Beispiele (unterstrichene Postleitzahlen und Zahlen mit Linien von benachbarten Feldern) aus der Testmenge ausgeschlossen wurden. Für die Fehlerrate von 2.9% wurde eine Erkennungsrate von 73.9% gemessen. Der in dieser Diplomarbeit vorgestellte Zahlenerkenner erreichte (allerdings ohne Ausschluss von Testbeispielen) für diese Fehlerrate eine Erkennungsrate von 63%.

Das in [SS95] vorgeschlagene Erkennungssystem basiert ebenfalls auf dem segmentierungsfreien Ansatz. Für die 495 Beispiele des Verzeichnisses *test/binzips* publizierten N.W. Strathy und C.Y. Suen eine Erkennungsrate von 56% für die Fehlerrate von 0.8%. Für die 435 Beispiele des Verzeichnisses *test/zipcodes* erreichte ihr System eine Erkennungsrate von 42% für die Fehlerrate von 1.2%. Die entsprechenden Erkennungsraten des in dieser Diplomarbeit implementierten Zahlenerkenners betragen für die *test/binzips* Testdaten 51% und für die *test/zipcodes* Testdaten 51%.

8.2.2 Vergleich für NIST SD3 Bilddaten

Im Gegensatz zu den Arbeiten, welche die CEDAR Datenbank verwendet haben, sind für die NIST SD3 Bilddaten mehr Arbeiten verfügbar, um die in dieser Diplomarbeit erzielten Erkennungsraten zu vergleichen.

Bis auf die Arbeit von J. Keeler und D.E. Rumelhart [KR92], die ebenfalls die Zahlen der NIST Formulare 'f1800' bis 'f1899' und 'f2000' - 'f2099' für Tests verwendet haben, sind die verwendeten Testdaten nirgends näher spezifiziert worden.

Publikation	Länge	Beispiele	Null	2.0%	1.0%	0.5%
[KR92]	2	1000	-	-	87.1 (0.9)	74.6 (0.4)
	3	1000	-	-	84.2 (0.9)	70.7 (0.4)
	4	1000	-	-	76.2 (0.8)	68.7 (0.3)
	5	1000	-	-	70.3 (0.7)	55.7 (0.3)
	6	1000	-	68.6 (1.4)	62.4 (0.6)	57.7 (0.3)
[LL95]	2	1000	-	96.5 (2.0)	95.2 (1.0)	94.5 (0.5)
	3	1000	-	92.1 (1.9)	88.0 (0.9)	82.6 (0.4)
	4	1000	-	84.3 (1.7)	80.7 (0.8)	75.6 (0.4)
	5	1000	-	81.3 (1.7)	78.6 (0.8)	70.7 (0.4)
	6	1000	-	77.4 (1.6)	70.5 (0.7)	43.8 (0.2)
[FGK95]	5	2000	83.1	69.6 (2.0)	64.3 (1.0)	61.4 (0.5)

Tabelle 8.2: Erkennungsraten publizierter Systeme für NIST Bilddaten

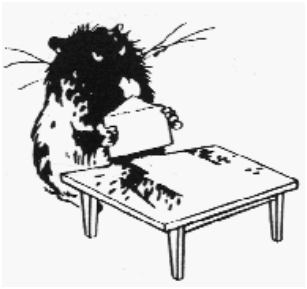
Tabelle 8.2 fasst die publizierten Erkennungsraten für verschiedene Fehlerraten von den Publikationen [KR92], [LL95] und [FGK95] zusammen. Sowohl der in [KR92] als auch der von S.W. Lee und E.J. Lee in [LL95] beschriebene Zahlenerkenner setzen Neuronale Netze zur Segmentierung und Erkennung der Zahlen ein (kontinuierlicher Ansatz). Das von A. Filatov, A. Gitis und I. Kil in [FGK95] publizierte System basiert auf dem segmentierungsfreien Ansatz und verwendet zur Segmentierung und Erkennung der Zahlen Subgraphisomorphismen.

Die Spalte 'Länge' der Tabelle 8.2 spezifiziert die Länge der Zahlen (Anzahl der enthaltenen Ziffern) der untersuchten Beispiele. Die Spalte 'Beispiele' enthält die Anzahl der getesteten Beispiele. In der Spalte 'Null' sind die publizierten Erkennungsraten für die Rückweisungsrate 0 angegeben und die Spalten '2%', '1%' und '0.5%' enthalten die Erkennungsraten für die entsprechend vorgegebenen Fehlerraten. Für Felder, die mit '-' markiert sind, fehlten die entsprechenden Angaben zu den Erkennungsraten.

Da die Autoren von [KR92] und [LL95] nicht die in Abschnitt 3.2 verwendeten Definitionen für die Fehlerrate verwendeten, konnten die Erkennungsraten nicht genau

für die in der Tabelle angegebenen Fehlerraten berechnet werden, die zugehörigen Fehlerraten sind deshalb in Klammern angegeben. Da für diese Fehlerraten häufig keine expliziten Erkennungsraten angegeben waren, mussten diese aus den publizierten Kurven (Fehlerrate zu Rückweisungsrate) herausgemessen werden. Die Erkennungsraten in Tabelle 8.2 können mit denjenigen in Tabelle 7.9 verglichen werden, falls die in Klammern angegebenen Fehlerraten mit der Bezeichnung der jeweiligen Spalte übereinstimmen. Andernfalls müssen diese Werte mit den Kurven in Abbildung 7.6 verglichen werden.

Das in dieser Arbeit vorgestellte Erkennungssystem erreicht für Zahlen ab der Länge 4 fast überall gleich hohe Erkennungsraten und für Zahlen ab der Länge 5 mehrheitlich höhere Erkennungsraten als die in Tabelle 8.2 zusammengefassten Ergebnisse. Generell kann festgestellt werden, dass der entwickelte Zahlenerkenner für kleine Rückweisungsraten und Zahlen mit vielen Ziffern im Vergleich zu anderen Publikationen höhere Erkennungsraten erreicht.



Kapitel 9

Schlussfolgerungen

Die in diesem Kapitel gezogenen Schlussfolgerungen beziehen sich auf den in dieser Arbeit vorgestellten Zahlenerkenner. Die Schlussfolgerungen für den Paarerkenner sind im letzten Abschnitt des Kapitels 5 dokumentiert und werden hier nicht mehr wiederholt.

Diese Diplomarbeit stellt ein System zur off-line Erkennung von handgeschriebenen Zahlen vor, das auf der Diplomarbeit von Dieter Niggeler basiert [Nig94]. Das System verbindet eine segmentierungsbasierte und eine segmentierungsfreie Methode zur Zahlenerkennung. Es basiert auf den vier spezialisierten Modulen 'Vorsegmentierung', 'Detektion Ziffern', 'Erkennung Teilbild', 'Kombination & Entscheidung' und einem Ziffernerkenner.

Der vorgestellte Zahlenerkenner wurde mit Bilddaten der CEDAR und der NIST SD3 Datenbank getestet. Auf beiden Datenbanken wurden sehr gute Erkennungsraten ohne Rückweisungen von Beispielen und ein gutes Rückweisungsverhalten für kleine Fehlerraten ($\leq 1\%$) erreicht. Die wichtigsten Erkenntnisse dieser Diplomarbeit sind in der folgende Aufstellung zusammengefasst.

- Durch die vorgeschlagene Kombination zweier verschiedener Segmentierungsverfahren lassen sich die Stärken beider Methoden nutzen, ohne dass ihre Schwächen voll zum Vorschein kommen.
- Das Module 'Vorsegmentierung' hat sich als sehr wirkungsvoll und zuverlässig erwiesen, da sich damit die meisten Beispiele korrekt in isolierte Ziffern segmentieren lassen. Da dieses Modul komplizierte Beispiele nur in Zifferngruppen unterteilen muss, können Segmentierungsfehler auf dieser Stufe der Erkennung fast vollständig vermieden werden.
- Die Detektion von isolierten Ziffern im Modul 'Detektion Ziffern' durch die Kombination eines Ziffernerkenners und einer Transitionsmessung hat sich ebenfalls als zuverlässig erwiesen.

- Die einfache Rückweisungsregel im Modul 'Kombination & Entscheidung' erlaubt es, das Erkennungssystem effizient auf eine vorgegebene Fehlerrate oder Rückweisungsrate einzustellen.
- Das Verhalten von Max [Gio93], der diese Dokumentation begleitet hat, muss als ausserordentlich beharrlich eingestuft werden.

Literaturverzeichnis

[Amm96] Roger Ammann. Rekonstruktion von Online-Information in der Handschrifterkennung. Master's thesis, Institiut für Informatik und angewandte Mathematik, Universität Bern, 1996.

[Bun92] Prof. H. Bunke. Vorlesungsskriptum Künstliche Intelligenz WS 92-93. Institut für Informatik und angewandte Mathematik, 1992.

[Bun95] Prof. H. Bunke. Vorlesungsskriptum Bildanalyse WS 95-96. Institut für Informatik und angewandte Mathematik, 1995.

[CDIP95] G. Congedo, G. Dimauro, S. Impedovo, and G. Pirlo. Segmentation of numeric strings. In *Proceedings of the third International Conference on Document Analysis and Recognition*, pages 1038–1041, Montréal, Canada, Aug. 14-16 1995.

[CGM93] T. Caesar, J. Gloger, and E. Mandler. Preprocessing and feature extraction for a handwriting recognition system. In *2nd IARP Int. Conf. on Document Analysis and Recognition*, pages 408–411, Tsukuba Science City, Oct. 1993.

[Cho57] C.K. Chow. An optimum character recognition system using decision functions. In *Institute of Radio Engineers (IRE) Transactions on Electronic Computers*, pages 247–254, Dec. 1957.

[Cho70] C.K. Chow. On optimum recognition error and reject tradeoff. In *IEEE Transactions on Information Theory*, volume IT-16, No 1, pages 41–46, Jan. 1970.

[Das90] Belur V. Dasarathy, editor. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamitos, California, 1990.

[Dev82] P. Devijver. *Pattern Recognition*. Prentice-Hall International, 1982.

[Dud76] Sahibsingh A. Dudani. The distance-weighted k-nearest-neighbour rule. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6(4):325–327, April 1976.

[FGK95] A. Filatov, A. Gitis, and I. Kil. Graph-based handwritten digit string recognition. In *Proceedings of the third International Conference on Document Analysis and Recognition*, pages 845–848, Montréal, Canada, Aug. 14-16 1995.

[Gio93] Pericle Luigi Giovannetti. *MAX*. Wilhelm Heyne Verlag GmbH & Co. KG, München, 1993.

[Gro94] Arnold Grossmann. Handwritten numeral recognition by a contour method. Technical Report IAM-PR-500, Institut für Informatik und angewandte Mathematik, Universität Bern, 1994.

[HB94] Thien M. Ha and H. Bunke. Handwritten numeral recognition by perturbation method. In *Proc. of the Fourth Int. Workshop on Frontiers of Handwriting Recognition*, pages 97–106, Taipei, Taiwan, Dec. 7-9 1994.

[Hul94] J.J. Hull. A database for handwritten text recognition research. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, volume 16, No. 5, pages 550–554, May 1994.

[KMS95] F. Kimura, Y. Miyake, and M. Shridhar. Handwritten zip code recognition using lexicon free word recognition. In *Proceedings of the third International Conference on Document Analysis and*, pages 906–910, Montréal, Canada, Aug. 14-16 1995.

[KR92] J. Keeler and D.E. Rumelhart. A self-organizing integrated segmentation and recognition neural network. In J.E. Moody, S.J. Hanson, and R.P. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 496–503, San Mateo, CA, Aug. 14-16 1992. Morgan Kaufmann.

[Kre91] Ulrich H.-G Kressel. The impact of the learning-set size in handwritten-digit recognition. In O. Simula T. Kohonen, K. Mäkisara and J. Kangas, editors, *Artificial Neural Networks*, pages 1685–1689. Elsevier Science Publishers B.V. (North-Holland), 1991.

[KRL91] J. D. Keeler, D. E. Rumelhart, and W. K. Leow. Integrated segmentation and recognition of hand-printed numerals. In D. S. Touretzky R. P. Lippmann, J. E. Moody, editor, *Advances in Neural Information Processing Systems*, volume 3, pages 557–563. Morgan Kaufmann, 1991.

[KS92] F. Kimura and M. Shridhar. Segmentation-recognition algorithm for zip code field recognition. *Machine Vision and Applications*, 5, No. 3:199–210, Summer 1992.

[LL95] S.W. Lee and E.J. Lee. Integrated segmentation and recognition of connected handwritten characters with recurrent neural network. In *Proceedings of the third International Conference on Document Analysis and Recognition*, pages 413–416, Montréal, Canada, Aug. 14-16 1995.

[Mar96] Urs-Viktor Marti. Einsatz Neuronaler netzwerke zur handschrifterkennung. Master's thesis, Institut für Informatik und angewandte Mathematik, Universität Bern, 1996.

[Nig94] Dieter Niggeler. Ein Erkennungssystem für Handgeschriebene Zahlen. Master's thesis, Institiut für Informatik und angewandte Mathematik, Universität Bern, 1994.

[NM92] H. Nishida and S. Mori. A model-based split-and-merge method for recognition and segmentation of character strings. In H. Bunke, editor, *Advances in Structural and Syntactic Pattern Recognition*, pages 300–309. World Scientific, 1992.

[Ots79] N. Otsu. A threshold selection method from gray-level histogram. *IEEE Trans.*, SMC-9:62–66, Jan. 1979.

[SNG92] P. L. Sparks, M. V. Nagendraprasad, and A. Gupta. An algorithm for segmenting handwritten numeral strings. In *Second Int. Conf. on Automation, Robotics, and Computer Vision*, pages CV–1.1.1 to CV–1.1.4, Singapore, Sept. 16-18 1992.

[SNL+92] Y. Suen, C. Nadal, R. Legault, T.A. Mai, and L. Lam. Computer recognition of unconstrained handwritten numerals. In T. Pavlidis and S. Mori, editors, *Proceedings of the IEEE*, pages 1162–1180, 1992. Special Issue on Optical Character Recognition.

[SS95] N.W. Strathy and C.Y. Suen. A new system for reading handwritten zip codes. In *Proceedings of the third International Conference on Document Analysis and Recognition*, pages 74–77, Montréal, Canada, Aug. 14-16 1995.

[Ste95] Peter Steiner. Zwei ausgewählte Probleme zur Offline-Erkennung von Handschrift. Master's thesis, Institut für Informatik und angewandte Mathematik, Universität Bern, 1995.

[WGJ+92] R.A. Wilkinson, J. Geist, S. Janet, P.J. Grother, C.J.C. Burges, R. Creecy, B. Hammond, J.J. Hull, N.W. Larsen, T.P. Vogl, and C.L. Wilson. The first census optical character recognition systems conference. Technical Report #NISTIR 4912, The U.S. Bureau of Census and the National Institute of Standards and Technology, Aug. 1992.