Inhaltsbasierter Bildzugriff mittels Statistischer Objekterkennung

-korrigierte Fassung-

Diplomarbeit am Lehrstuhl für Informatik VI der RWTH Aachen

Prof. Dr.-Ing. H. Ney

vorgelegt von:

Mark Oliver Güld
Matrikelnummer 196 807

Gutachter:

Prof. Dr.-Ing. H. Ney
Prof. Dr. J. Hromkovič

Betreuer:

Dipl.-Inform. J. Dahmen

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

Aachen, den 31. Juli 2000

Mark Oliver Güld

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Danksagung

Dank gilt an dieser Stelle den Personen, deren Hilfe durch Wort und Tat die Fertigstellung dieser Arbeit ermöglicht hat: Herrn Prof. Dr. Ney, an dessen Lehrstuhl diese Diplomarbeit entstand, meinem Betreuer Jörg Dahmen und den Mitstudenten Alexander Crämer, Daniel Keysers, Ralf Perrey und Thomas Theiner, durch die nicht nur ein produktives, sondern auch sehr angenehmes Arbeitsklima entstand.

Kapitel 1

Einleitung und Ziele dieser Arbeit

In der Vergangenheit haben sich statistische Klassifikatoren auf dem Gebiet der Spracherkennung bewährt [26]. In dieser Arbeit soll ihre Eignung für die Bilderkennung untersucht werden. Im Rahmen dessen wird zunächst der Aufbau eines Erkennungssystems für Einzelobjekte entwickelt, welches sich durch folgende Charakteristika auszeichnet:

- allgemeine Verwendbarkeit zur (kontextfreien) Mustererkennung,
- statistische Modellierung des Datenmaterials,
- in der Grundversion Verzicht auf problemspezifisches Wissen,
- wohldefinierte Punkte zur Integration von problemspezifischem Wissen,
- Konkurrenzfähigkeit mit anderen 'State-of-the-Art'-Ansätzen wie Support-Vektor-Maschinen und Neuronale Netzen.

Der Aufbau des Erkennungssystems wird in Kapitel 2 beschrieben. Im Rahmen dessen werden die auf der Datensammlung USPS erzielten Ergebnisse diskutiert sowie (kurz) konkurrierende Klassifikatoren vorgestellt.

Danach soll die Tauglichkeit des Erkennungssystems zur allgemeineren Lokalisierung und Klassifikation von Objekten in Bildmaterial untersucht werden. Hierbei liegt das Augenmerk auf einem segmentierungsfreien Ansatz, d.h. es wird nicht versucht, in einem vorgeschalteten Schritt die Objekte einer Testzene auszuschneiden und diese zu klassifizieren, sondern die Szene wird als ganzes untersucht. Es soll untersucht werden, welche neuen Probleme bei dieser Aufgabe auftreten und wie diese teilweise gelöst werden können. Dies wird in Kapitel 3 beschrieben, wobei auch hier andere Ansätze vorgestellt werden sollen. Praktische Untersuchungen des Systems finden auf modifizierten Daten der Datensammlung USPS sowie auf der Datensammlung COIL-20 statt. Ziel ist es, ein System zu entwickeln, mit dem Bilder anhand der Objekte, die sie enthalten, charakterisiert werden können. Diese Bildbeschreibungen können dann - in einer Datenbank gespeichert - benutzt werden, um Bildmaterial zu archivieren und wiederzufinden. Dies wird mit *content based Image Retrieval* (inhaltsbasierter Bildzugriff) bezeichnet. Vorteil einer automatischen Indizierung von Bildmaterial ist neben der offensichtlichen Arbeitsersparnis auch die nicht zu unterschätzende Reproduzierbarkeit von Ergebnissen. Bildindizierungen durch Menschen erfolgen unter Umständen aufgrund unterschiedlicher Sichtweisen, je nachdem, welcher Aspekt den Betrachter gerade interessiert. Das Indizierungssystem arbeitet hingegen (auch im Fall falscher Indizierung) stets nachvollziehbar. Das Finden von Objekten in beliebigen Testszenen findet ferner häufig in Zusammenhang mit Roboter-Wahrnehmung sowie Gesichts- und Gestenerkennung statt, weshalb generell ein hohes Interesse in diesem Forschungbereich besteht.

Die im Rahmen dieser Arbeit durchgeführten Untersuchungen und Ergebnisse sind in folgende Publikationen eingeflossen:

- J. Dahmen, K. Beulen, M.O. Güld, H. Ney: *A Mixture Density Approach to Object Recognition for Image Retrieval*, 6th International RIAO Conference on Content-Based Multimedia Information Access, Seite 1632-1647, Paris, Frankreich, April 2000.
- J. Dahmen, D. Keysers, M.O. Güld, H. Ney: *Invariant Image Object Recognition*, 15th International Conference on Pattern Recognition, Barcelona, Spanien, September 2000.
- M. Schreckenberg, J. Dahmen, M.O. Güld, G. Schummers, D. Meyer-Ebrecht, H. Ney: *Automatische Endokarderkennung in 3D mit approximierenden Thin-Plate-Spline Modellen unter Einsatz von Gauß'schen Mischverteilungs-Modellen für die lokale Klassifikation*, Bildverarbeitung für die Medizin 2000, S. 351-355, März 2000.

Kapitel 2

Statistische Klassifikation von Einzelobjekten

2.1 Einleitung

Grundlegendes Problem der Mustererkennung ist es, eine Zuordnung zu bestimmen, die ein Signal S automatisch einer von K Klassen zuordnet. Die Konstruktion der Zuordnung soll anhand einer Menge von Signalen, deren Klassenzugehörigkeit bekannt ist, - den sogenannten Trainingsdaten - stattfinden.

Zunächst ist hierbei zu überlegen, wie ein Signal S (in diesem Fall ein Bild) beschrieben werden soll. Die Darstellung von S wird *Merkmalsvektor* $x \in \mathbb{R}^D$ genannt. Die Komponenten von x quantifizieren die Eigenschaften des Signals im sogenannten *Merkmalsraum* $\mathbb{R}^D$. Der Übergang von S zu $x \in \mathbb{R}^D$ wird *Merkmalsanalyse* genannt. In dieser Arbeit, die sich mit der Klassifikation von Bildmaterial beschäftigt, wird die *erscheinungsbasierte Merkmalsanalyse* verwendet, d.h. die für die Klassifikation verwendete Repräsentation des Bildsignals besteht zunächst aus den Werten der Grauwertfunktion $I(i, j)$. Später wird motiviert und dargestellt, wie sich diese -sehr große- Anzahl von Merkmalen auf wenige, dafür möglichst aussagekräftige reduzieren läßt.
Mit der Wahl einer Merkmalsextraktion kann das Klassifikationsproblem folgendermaßen formuliert werden:

$$\begin{aligned} \text{Bestimme } r : \mathbb{R}^D &\rightarrow \{1, \ldots, K\} \\ r(x) &= k \quad \Leftrightarrow \quad \text{"Merkmalsvektor } x \text{ wird Klasse } k \text{ zugeordnet"} \end{aligned}$$

Die Klassen seien mit $A_1, \ldots, A_K$, $A_k \subseteq \mathbb{R}^D$ bezeichnet, d.h. die Menge A_k enthält alle Merkmalsvektoren der Klasse k. Hilfsmittel zur Konstruktion von r sind die Trainingsdaten. Dabei handelt es sich um eine Menge von Merkmalsvektoren, deren Klassenzugehörigkeit bekannt ist:

$$\{x_{1k}, \ldots, x_{N_k k}\} \subseteq A_k, \quad x_{ik} \in \mathbb{R}^D,\ i = 1, \ldots, N_k \quad k = 1, \ldots, K$$

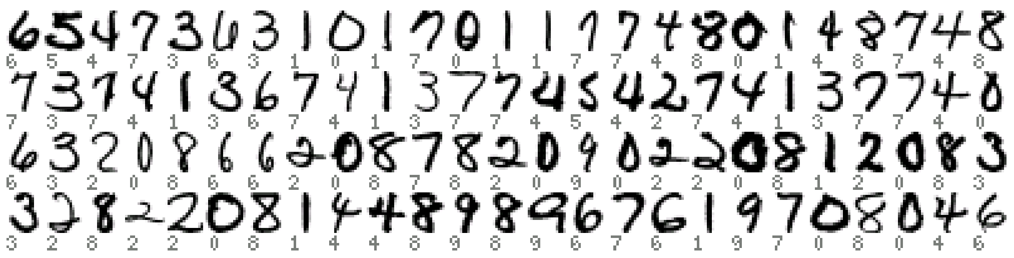

Abbildung 2.1: Die ersten 100 Trainingsdaten des USPS-Korpus

Wünschenswert ist hierbei, daß r insbesondere auch Merkmalsvektoren $x \in I\!R^D$ korrekt klassifiziert, die nicht unter den Trainingsdaten zu finden sind, also eine möglichst gute Fähigkeit zur Generalisierung.

2.2 Die Datensammlung USPS und Stand der Technik

Der Datenkorpus, auf dem die in diesem Kapitel beschriebenen Methoden evaluiert werden, ist die mit United States Postal Service (USPS) bezeichnete Datensammlung isolierter, handgeschriebener Ziffern, die aus Postleitzahlen auf Briefumschlägen extrahiert wurden. Der Korpus ist unterteilt in 7291 Trainingsdaten und 2007 Testdaten, wobei jedes Datum ein 16x16 Pixel großes Grauwertbild mit 256 Intensitätswerten ist. Die ersten 100 Ziffern des Trainingskorpus sind in Abbildung 2.1 dargestellt.

Der Korpus ist frei verfügbar[1] und wird in vielen Publikationen als Datenbasis zur Evaluation benutzt. Bisher publizierte Ergebnisse zu diesem Korpus sind in Tabelle 2.1 dargestellt. Die menschliche Fehlerrate ist mit 2.5% recht hoch, was auf ein schwieriges Erkennungsproblem hindeutet. Ferner stellt die relativ geringe Anzahl an Trainingsdaten eine Schwierigkeit insbesondere für statistische Ansätze dar, was später detailliert erläutert werden soll.

Einige der konkurrierenden Klassifikatoren, nämlich neuronale Netze und die Support-Vektor-Maschine, werden in Abschnitt 2.7.2 kurz vorgestellt.

2.3 Bayes'sche Entscheidungsregel

Ziel der Mustererkennung ist es, eine Entscheidungsfunktion $r : I\!R^D \to \{1, \ldots, K\}, x \mapsto r(x)$ zu finden, die mit möglichst geringer Fehlerrate arbeitet. Die Bewertung einer Entscheidungsfunktion [25, S. 24] erfolgt, indem die Kosten der von r getroffenen Entscheidungen über alle Merkmalsvektoren betrachtet werden:

$$R = \int_x dx \sum_{c=1}^{K} p(c|x) \cdot L[c, r(x)] \tag{2.1}$$

wobei $L[c, k]$ die lokalen Kosten einer für $x \in I\!R^D$ getroffenen Entscheidung für Klasse k sind:

[1] `ftp://ftp.kyb.tuebingen.mpg.de/pub/bs/data`

Tabelle 2.1: Bisher publizierte Ergebnisse für den USPS Datenkorpus

Methode/Publikation	Fehlerrate in %
Mensch [35]	2.5
Neuronales Netz, LeNet1 [16]	*4.5
NN, Euklidische Distanz [35]	**5.9
NN, Tangentendistanz [35]	**2.6
Support-Vektor-Maschine [32]	4.2
Support-Vektor-Maschine mit virtuellen Support-Vektoren [32]	3.2

*: Das USPS-Testset wurde um 700 maschinengedruckte Ziffern erweitert. Die Autoren geben eine Fehlerrate von 3.4% an, erwähnen jedoch, daß alle Klassifikationsfehler bei handgeschriebenen Ziffern auftraten, wodurch sich für das USPS-Testset die angegebene Fehlerrate von 4.5% ergibt. Das USPS-Trainingsset wurde um 2.549 maschinengedruckte Ziffern erweitert. Die Ergebnisse sind somit nur eingeschränkt vergleichbar.

**: Das USPS-Trainingsset wurde um 2.418 maschinengedruckte Ziffern erweitert. Die Ergebnisse sind somit nur eingeschränkt vergleichbar.

$$L[c,k] = \begin{cases} 0 & c = k \\ 1 & c \neq k \end{cases} \tag{2.2}$$

Die Kosten werden minimal, d.h. die Klassifikationsentscheidung ist optimal, falls als Entscheidungsfunktion

$$r(x) := \underset{k=1,\dots,K}{\operatorname{argmin}} \left\{ \sum_{c=1}^{K} p(c|x) \cdot L[c,k] \right\} \tag{2.3}$$

gewählt wird [25]. Durch weiteres Umformen ergibt sich

$$r(x) = \underset{k=1,\dots,K}{\operatorname{argmax}} \; \{p(k|x)\} \tag{2.4}$$

Die optimale Entscheidung besteht demnach darin, den Merkmalsvektor x stets der Klasse mit maximaler a-posteriori-Wahrscheinlichkeit $p(k|x)$ zuzuordnen. In der Realität ist diese Verteilung allerdings unbekannt und muß auf geeignete Art und Weise modelliert werden.

Mit der Umformung

$$\begin{aligned} \underset{k}{\operatorname{argmax}} \; \{p(k|x)\} &= \underset{k}{\operatorname{argmax}} \; \{p(x) \cdot p(k|x)\} = \underset{k}{\operatorname{argmax}} \; \{p(x,k)\} \\ &= \underset{k}{\operatorname{argmax}} \; \{p(k) \cdot p(x|k)\} \end{aligned} \tag{2.5}$$

ergibt sich eine andere Darstellung der Entscheidungsregel r, welche die posterior-Wahrscheinlichkeiten als Produkt der a-priori-Wahrscheinlichkeit $p(k)$ für Klasse k und klassenbedingter Wahrscheinlichkeit $p(x|k)$ modelliert. Insgesamt ergibt sich für das entstehende Erkennungssystem die in

Abbildung 2.2 dargestellte Struktur. Die Vorverarbeitung, der Schritt zwischen Merkmalsanalyse und Klassifikation, soll später motiviert und erläutert werden.

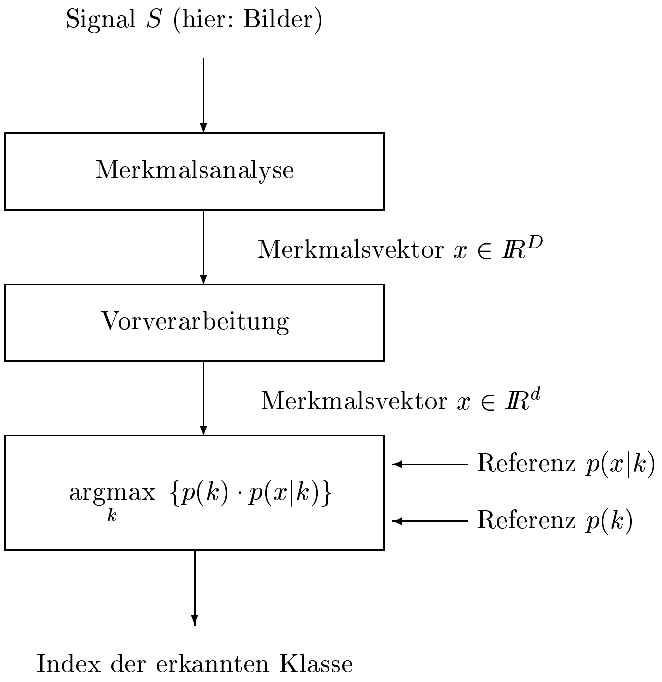

Abbildung 2.2: Aufbau eines statistischen Klassifikators für Einzelobjekte [25, S. 7]

Für $p(k)$ kann die auf den Trainingsdaten empirisch ermittelte a-priori-Klassenwahrscheinlichkeit gewählt werden. Da der untersuchte Korpus USPS aber isolierte Ziffern ohne weitere Nebenbedingungen enthält, wird für diese Aufgabe eine Gleichverteilung angenommen, also $p(k) = \frac{1}{K}$, $k = 1, \ldots, K$.

2.4 Modellierung der klassenbedingten Wahrscheinlichkeit

Ausgehend von der in Gleichung (2.5) gewählten Darstellung der posterior-Wahrscheinlichkeit soll nun die Verteilung $p(x|k)$ modelliert werden.

2.4.1 Gauß'sche Mischverteilungen

Zur Modellierung der klassenbedingten Wahrscheinlichkeiten $p(x|k)$ können Gauss'sche Mischverteilungen benutzt werden. Diese ergeben sich als gewichtete Summe Gauss'scher Normalverteilungen (im folgenden auch *Dichten* genannt) für Merkmalsvektoren $x \in \mathbb{R}^D$:

$$p(x|k) = \sum_{i=1}^{I_k} c_{ki} \cdot \mathcal{N}(x|\mu_{ki}, \Sigma_{ki}) \tag{2.6}$$

wobei $c_{ki} \geq 0$ Gewichtungsfaktoren der Einzelverteilungen $\mathcal{N}(x|\mu_{ki}, \Sigma_{ki})$ sind und mit $\sum_i c_{ki} = 1$ ferner die Normierungsbedingung für Wahrscheinlichkeitsverteilungen gewährleisten.

Eine GAUSS'sche Einzelverteilung $\mathcal{N}(x|\mu, \Sigma)$ ist definiert als

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^D \det \Sigma}} \exp\left[-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right] \tag{2.7}$$

Jede der Einzelverteilungen $\mathcal{N}(x|\mu_{ki}, \Sigma_{ki})$ in (2.6) benutzt einen eigenen, d.h. klassen- und dichtenspezifischen Mittelwert μ_{ki} sowie eine klassen- und dichtenspezifische Kovarianzmatrix Σ_{ki}. GAUSS'sche Mischverteilungen haben die Eigenschaft, daß durch sie jede Verteilung beliebig genau approximiert werden kann [25, S. 93].
Somit ergibt sich ein Modell, mit dem multimodale multivariate Wahrscheinlichkeitsverteilungen, d.h. mehrdimensionale Wahrscheinlichkeitsverteilungen mit mehreren Maxima, dargestellt werden können. Die Frage ist nun, wie die Verteilungsparameter μ_{ki} und Σ_{ki} ermittelt werden können.

Maximum-Likelihood-Parameterschätzung

Gesucht ist eine optimale Wahl der Verteilungsparameter, d.h. die beste Erklärung der vorhandenen Trainingsdaten $\{x_{1k}, \ldots, x_{N_k k}\}$ durch das Modell $p(x|k)$. Dies kann mit Hilfe der *Likelihood* ausgedrückt werden [25, S. 50ff], die für einen Parametersatz ϑ_k die Wahrscheinlichkeit der gegebenen Trainingsdaten für das Modell $p(x|k, \vartheta_k)$ angibt:

$$\vartheta_k \mapsto \prod_{n=1}^{N_k} p(x_{nk}|k, \vartheta_k) \tag{2.8}$$

Bei einer GAUSS'schen Einzelverteilung besteht der Parametersatz aus klassenspezifischem Mittelwert und klassenspezifischer Kovarianzmatrix, also $\vartheta_k = (\mu_k, \Sigma_k)$. Gesucht ist demnach

$$(\hat{\mu}_k, \hat{\Sigma}_k) = \operatorname*{argmax}_{\mu_k, \Sigma_k} \left\{ \prod_{n=1}^{N_k} \mathcal{N}(x_{nk}|\mu_k, \Sigma_k) \right\} \tag{2.9}$$

Durch Differenzieren und Nullsetzen ergeben sich als beste Schätzer [25]

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} x_{nk} \tag{2.10}$$

$$\hat{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N_k} (x-\hat{\mu}_k)(x-\hat{\mu}_k)^T \tag{2.11}$$

d.h. der empirische Mittelwert und die empirische Kovarianzmatrix.

Eine Maximum-Likelihood-Schätzung für GAUSS'sche Mischverteilungen erweist sich als schwieriger, da der Zusammenhang zwischen einem Trainingsvektor x_{nk} und den Dichten der Mischverteilungen nicht beobachtbar ist, d.h. eine verborgene Variable existiert.

Der Expectation-Maximization-Algorithmus

Zum Training der Parameter für GAUSS'sche Mischverteilungen (2.6) wird in dieser Arbeit der Expectation-Maximization-Algorithmus (kurz EM-Algorithmus) verwendet. Hierbei wird die unbekannte Zuordnung eines Trainingsdatums x_{nk}, $n = 1, \ldots, N_k$ zu den Dichten $\mathcal{N}(x_{nk}|\mu_{ki}, \Sigma_{ki})$, $i = 1, \ldots, I_k$ mit einer verdeckten Variable y modelliert [25, S. 96ff]. Für die Konstruktion der Mischverteilung einer Klasse k werden lediglich die Trainingsdaten zu dieser Klasse benutzt, d.h. $\{x_{nk},\ n = 1, \ldots, N_k\}$. Im folgenden wird daher der Klassenindex k weggelassen. Allgemein ergibt sich bei Modellierung mit verborgener Variable y:

$$p(x_n|c_i, \mu_i, \Sigma_i) = \sum_y p(x_n, y|c_i, \mu_i, \Sigma_i) \tag{2.12}$$

$$= \sum_y p(y|c_i, \mu_i, \Sigma_i) \cdot p(x_n|y, c_i, \mu_i, \Sigma_i) \tag{2.13}$$

Bei GAUSS'schen Mischverteilungen modelliert die verdeckte Variable y die nicht beobachtbare Zuordung eines Trainingsvektors zu den einzelnen Dichten seiner Klasse, d.h. $y \equiv i$. Somit ergibt sich

$$p(x_n|c_i, \mu_i, \Sigma_i) = \sum_i \underbrace{p(i|c_i, \mu_i, \Sigma_i)}_{\equiv c_i} \cdot \mathcal{N}(x_n|\mu_i, \Sigma_i) \tag{2.14}$$

$$= \sum_i c_i \cdot \mathcal{N}(x_n|\mu_i, \Sigma_i) \tag{2.15}$$

Der EM-Algorithmus erreicht eine iterative Verbesserung der Parametereinstellung $\lambda_i = \{c_i, \mu_i, \Sigma_i\}$, $i = 1, \ldots, I$, indem zunächst die Differenz zwischen den Likelihoods zweier Parametersätze $\lambda, \bar{\lambda}$ berechnet (Expectation) und optimiert (Maximization) wird. Die Anwendung des EM-Algorithmus zur Parameterschätzung für GAUSS'sche Mischverteilungen wird in [10] dargestellt und ergibt folgende Reestimationsformeln:

$$p(i|x_n, \lambda_i) = \frac{c_i \cdot \mathcal{N}(x_n|\mu_i, \Sigma_i)}{\sum_{i'=1}^{I} c_{i'} \cdot \mathcal{N}(x_n|\mu_{i'}, \Sigma_{i'})} \tag{2.16}$$

$$\gamma_i(n) = \frac{p(i|x_n, \lambda_i)}{\sum_{n'=1}^{N} p(i|x_{n'}, \lambda_i)} \tag{2.17}$$

$$\bar{c}_i = \frac{1}{N} \sum_{n=1}^{N} p(i|x_n, \lambda_i) \tag{2.18}$$

$$\bar{\mu}_i = \sum_{n=1}^{N} \gamma_i(n) \cdot x_n \tag{2.19}$$

$$\bar{\Sigma}_i = \sum_{n=1}^{N} \gamma_i(n) \cdot (x_n - \bar{\mu}_i)(x_n - \bar{\mu}_i)^T \tag{2.20}$$

Nach jeder Iteration werden die soeben neu geschätzten Parameter als Grundlage für den nächsten Iterationsschritt genommen, d.h. man setzt $c_{ki} = \bar{c}_{ki}$, $\mu_{ki} = \bar{\mu}_{ki}$ und $\Sigma_{ki} = \bar{\Sigma}_{ki}$.

Eine Schwierigkeit besteht in der Wahl der Startwerte für c_{ki}, μ_{ki} und Σ_{ki}, insbesondere für eine hohe Dichtenanzahl. Anstatt mit einer festen Anzahl von Dichten zu arbeiten, wird von dem in dieser Arbeit implementierten EM-Algorithmus zunächst eine Einzelverteilung pro Klasse geschätzt, wie in Abschnitt 2.4.1 dargestellt. Diese wird danach sukzessive geteilt, bis die gewünschte Anzahl von Dichten erreicht ist. Zwischen Teilungen werden jeweils mehrere Iterationen der EM-Schätzung ausgeführt. Das Teilen einer Dichte erfolgt, indem jeweils der Mittelwertvektor der Dichte ‚gestört' wird, typischerweise in Richtung der maximalen Varianz der betrachteten Dichte. Dieses Vorgehen orientiert sich an einem von Linde, Buzo und Gray in [19] vorgestellten Verfahren.

Ein Clustering-Algorithmus auf der Basis Gauß'scher Mischverteilungen

Die mit dem EM-Algorithmus trainierten Verteilungsparameter können benutzt werden, um einen Clustering-Algorithmus für die Trainingsdaten zu gewinnen: Man betrachtet jede Dichte der Mischverteilung als Verteilungsfunktion für einen Cluster. Die Zuordnung eines Trainingsdatums x_{nk} zu einem Cluster der Klasse k erfolgt durch

$$x_{nk} \mapsto \underset{i=1,\ldots,I_k}{\operatorname{argmax}} \{c_i \cdot \mathcal{N}(x|\mu_{ki}, \Sigma_{ki})\} \tag{2.21}$$

Hierdurch werden die Trainingsdaten in $\sum_k I_k$ viele Cluster unterteilt. Abbildung 2.3 zeigt die Mittelwerte einer Unterteilung der USPS-Trainingsdaten in 40 Cluster (4 pro Klasse). Durch die Mittelwertbildung erscheinen Bildregionen, in denen hohe Variabilität innerhalb des Clusters herrscht, verwaschen. Gut zu erkennen ist die Unterteilung der Klassen nach verschiedenen typischen Schreibweisen für die jeweilige Ziffer.

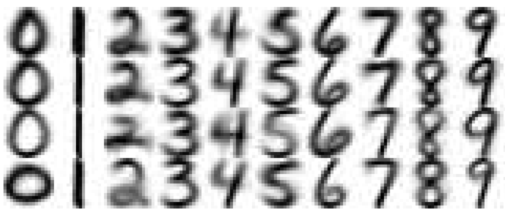

Abbildung 2.3: Unterteilung des USPS-Trainingskorpus in 40 Cluster. Dargestellt sind die Cluster-Mittelwerte. Jeder Cluster kann als *Pseudoklasse* betrachtet werden.

Das Problem der zuverlässigen Parameterschätzung

Die Modellierung der klassenbedingten Wahrscheinlichkeit als Gauss'sche Mischverteilung nach Gleichung (2.6) erfordert das Schätzen von

$$\sum_{k=1}^{K} \sum_{i=1}^{I_k} \left(\underbrace{D}_{\mu_{ki}} + \underbrace{\frac{D(D+1)}{2}}_{\Sigma_{ki}} \right) \overset{I_k = I\,\forall k}{=} K \cdot I \cdot \frac{1}{2}(D^2 + 3D) \tag{2.22}$$

Parametern (die Kovarianzmatrizen Σ_{ki} sind symmetrisch). Insbesondere ist die dichtenabhängige Kovarianzmatrix in Gleichung (2.7) bzw. (2.6) zu invertieren, was zu Schwierigkeiten führen kann.

Die empirische Kovarianzmatrix entsteht als normierte Summe äußerer Produkte von Vektoren mit sich selbst. Damit Σ invertierbar ist (d.h. vollen Rang besitzt), sind demnach mindestens D Vektoren pro Dichte notwendig, wodurch auf dem USPS-Korpus bei Verwendung klassen- und dichtespezifischer Kovarianzmatrizen und Beibehaltung aller 256 Merkmale nur etwa 3 Dichten pro Klasse schätzbar wären ($3 \cdot 256 \cdot 10 \approx 7291$). Dies approximiert die Verteilung der klassenbedingten Wahrscheinlichkeiten $p(k|x)$ nur unzureichend. In der Praxis fordert man für eine zuverlässige Schätzung von Σ sogar die gegenüber D zehn- bis hundertfache Anzahl von Trainingsvektoren. In [25, S.54] wird als weiteres Problem neben obigem die Unterschätzung der Varianz, bedingt durch eine zu geringe Variabilität der Lernstichprobe, genannt.

Grundsätzlich ergeben sich zumindest 3 Möglichkeiten, die Anzahl zu schätzender Parameter zu verringern, bzw. diese zuverlässiger zu schätzen:

1. Die Merkmalsreduktion, d.h. die Verkleinerung von D.
2. Die Beschaffung von mehr Trainingsmaterial. Dieser Schritt soll automatisiert ablaufen. Die zusätzlich erzeugten Daten sollen allein aus dem vorhandenen Trainingskorpus konstruiert werden.
3. Die Vereinfachung des statistischen Modells.

Punkt 1 wird in Abschnitt 2.5 untersucht werden, Punkt 2 in Abschnitt 2.6. Zunächst aber soll der dritte Punkt beleuchtet werden.

Die problematische Schätzung der Kovarianzmatrix kann dadurch umgangen werden, daß nur mit diagonalen Σ_{ki} gearbeitet wird, d.h. es werden nur noch Varianzvektoren $\sigma_{ki}^2 \in I\!R^D$ geschätzt. Dies bedeutet, daß die einzelnen Merkmale (also die Bildpunkte) als unkorreliert, d.h. statistisch unabhängig voneinander, angesehen werden (was eine recht starke Vereinfachung darstellt, die sich aber durch die erzielten Ergebnisse rechtfertigen läßt). Die Formulierung einer Einzelverteilung gemäß Gleichung (2.7) ändert sich dann zu

$$\mathcal{N}(x|\mu_k, \sigma_k^2) = \frac{1}{\prod_{d=1}^{D} \sqrt{2\pi\sigma_{kd}^2}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_d - \mu_{kd}}{\sigma_{kd}}\right)^2\right] \tag{2.23}$$

Durch Dekorrelation der Merkmale vor dem Training kann gehofft werden, den Informationsverlust dieser Maßnahme zu reduzieren. Ein Verfahren hierzu ist die in Abschnitt 2.5.3 beschriebene Whitening-Transformation.

In [8] wird ein Mittelweg zwischen der Benutzung voller Kovarianzmatrizen und der Benutzung diagonaler Kovarianzmatrizen dargestellt. Die Anzahl der freien Parameter für die Kovarianzmatrix kann reduziert werden, indem beim Schätzen dieser lediglich Korrelationen von Pixeln mit Pixeln seiner Nachbarschaft betrachtet werden. Daraus resultiert eine besondere Struktur von Σ bzw. Σ^{-1}. Eine weitere Möglichkeit zur Modellvereinfachung ergibt sich dadurch, nicht mit dichten-und klassenspezifischen Varianzen zu arbeiten, sondern lediglich mit klassenspezifischen oder sogar mit nur einer globalen Varianz. Dieses Vorgehen wird im folgenden mit *Varianzpooling* bezeichnet. Somit ergibt sich folgende schrittweise Vereinfachung:

$$
\begin{aligned}
p(x|k) &\overset{Modell}{=} \sum_{i=1}^{I_k} c_{ki} \cdot \mathcal{N}(x|\mu_{ki}, \Sigma_{ki}) && (2.24)\\
&\rightarrow \sum_{i=1}^{I_k} c_{ki} \cdot \mathcal{N}(x|\mu_{ki}, \sigma_{ki}^2) && (2.25)\\
&\rightarrow \sum_{i=1}^{I_k} c_{ki} \cdot \mathcal{N}(x|\mu_{ki}, \sigma_k^2), \text{ d.h. } \sigma_{ki}^2 = \sigma_k^2 \quad \forall i = 1, \ldots, I_k && (2.26)\\
&\rightarrow \sum_{i=1}^{I_k} c_{ki} \cdot \mathcal{N}(x|\mu_{ki}, \sigma^2), \quad \text{d.h. } \sigma_{ki}^2 = \sigma^2 \quad \forall k = 1, \ldots, K, \forall i = 1, \ldots, I_k && (2.27)
\end{aligned}
$$

Schritt (2.25) stellt den Übergang zu diagonalen Kovarianzmatrizen dar, d.h. zu voneinander unabhängigen Merkmalen. Schritt (2.26) bedeutet den Einsatz von klassenspezifischem Varianzpooling und schließlich Schritt (2.27) den Einsatz von globalem Varianzpooling.

Anmerkung: Im Falle eines global gepoolten Varianzvektors $\sigma^2 \in I\!R^D$ entsteht bei vor dem Training durchgeführter Whitening-Transformation (siehe Abschnitt 2.5.3, hinreichend ist die Diagonalisierung der within-class-scatter-Matrix S_w) für Einzelverteilungen kein Informationsverlust, da der gepoolte Varianzvektor σ^2 genau den Diagonaleneinträgen von S_w entspricht.

2.4.2 Kernel Densities

Eine weitere Möglichkeit, $p(x|k)$ zu modellieren, sind *Kernel Densities* (auch *Parzen Densities* oder *Parzen windows* [25, S. 82ff]). Prinzipiell wird $p(x|k)$ als Mischverteilung von N_k vielen Einzelverteilungen $\varphi_k(x)$ modelliert, d.h. jedes Trainingsdatum $x_{nk}, n = 1, \ldots, N_k$ aus Klasse k definiert eine Einzelverteilung. Es entsteht eine Art ‚verschmiertes' Histogramm. Als Einzelverteilung können beispielsweise multivariate Normalverteilungen mit unabhängigen Komponenten, d.h. Dichten mit Mittelwerten $x_{nk} \in I\!R^D$ und klassenspezifischer Varianz $\sigma_k^2 \in I\!R^D$ verwendet werden. Somit ergibt sich

$$
\begin{aligned}
p(x|k) &\overset{Modell}{=} \frac{1}{N_k} \sum_{n=1}^{N_k} \varphi_k(x - x_{nk}) \text{ , wobei} && (2.28)\\
\varphi_k(x) &= \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi\sigma_{kd}^2}} \exp\left[-\frac{1}{2}\left(\frac{x_d}{\sigma_{kd}}\right)^2\right], \quad x = (x_1, \ldots, x_D) \in I\!R^D && (2.29)
\end{aligned}
$$

Bei der Implementierung innerhalb dieser Arbeit wurde auch hier globales Varianzpooling benutzt, d.h. $\sigma_k^2 = \sigma^2 \, \forall k = 1, \ldots, K$.

2.4.3 Nearest Neighbour Entscheidungsregel

Die *nearest neighbour decision rule* oder kurz NN-Regel kann als Sonderfall von Kernel Densities hergeleitet werden, indem in (2.28) mit Maximum-Approximation statt der Summe gearbeitet wird [25, S.85ff]. In der vorliegenden Arbeit wird zusätzlich mit unabhängigen Komponenten und konstanter Varianz gearbeitet, d.h. $\Sigma = I$. Es wird diejenige Klasse gewählt, zu welcher der dem Testdatum $x \in I\!R^D$ nächste Nachbar aus den Trainingsdaten x_{nk} gehört.

$$r(x) = \operatorname*{argmin}_{k} \left\{ \min_{n=1,\ldots,N_k} ||x - x_{nk}|| \right\} \tag{2.30}$$

Als Abstandsmaß kann beispielsweise die Euklidische Distanz $||.||_2$ gewählt werden.

2.5 Merkmalsreduktion

Ziel der Merkmalsreduktion soll es sein, die Anzahl der Merkmale und damit die Anzahl der zu schätzenden Verteilungsparameter ‚sinnvoll' zu verringern, d.h. derart, daß für die Daten ‚typische' Eigenschaften im reduzierten Datenmaterial erhalten bleiben. Motivation hierbei ist die in Abschnitt 2.4.1 dargestellte Notwendigkeit, die Anzahl der Verteilungsparameter im statistischen Modell zu verringern. Die Merkmalsreduktion ist hierbei einer von drei in dieser Arbeit diskutierten Punkten (siehe Seite 20).

Betrachtet man die BAYES'sche Entscheidungsregel, sind beispielsweise die posterior-Wahrscheinlichkeiten $p(k|x)$, $k = 1, \ldots, K$ hinreichend, um eine optimale Entscheidung zu treffen. Hier reichen also statt D Merkmalen K (bzw. $K-1$, da $\sum_{k=1}^{K} p(k|x) = 1$) Merkmale aus. Allerdings sind diese Wahrscheinlichkeiten in der Realität unbekannt.

Prinzipiell muß stets davon ausgegangen werden, daß durch die Reduktion von D ein Informationsverlust eintritt, d.h. die potentielle Erkennungsleistung verringert wird [25, S. 129]. Es ist also ein Mittelweg zwischen Informationsverlust (sinkende theoretische Erkennungsleistung) und dadurch verbesserter Parameterschätzung im statistischen Modell (steigende Erkennungsleistung durch bessere Approximation) zu finden.

2.5.1 Die Karhunen-Loève-Transformation (KLT)

Ansatz der Karhunen-Loève-Transformation (auch Hauptkomponentenanalyse, engl. principal component analysis, PCA) ist es, für eine Menge von Vektoren, von der eine Stichprobe $x_1, \ldots, x_N \in I\!R^D$ gegeben ist, eine Darstellung mit weniger Komponenten zu finden, so daß der eintretende Informationsverlust möglichst gering ist. Folgende Darstellung ist [34, S. 114ff], entnommen. Der Informationsverlust beim Übergang von $x \in I\!R^D$ zu einem reduzierten Merkmalsvektor $y = (y_1, \ldots, y_d) \in I\!R^d$ wird hierbei als quadrierter Rekonstruktionsfehler $\epsilon(\tilde{x})$ gemessen, wobei $\tilde{x} \in I\!R^D$ der aus y rekonstruierte Vektor ist.

Sei Φ eine Matrix, deren Zeilenvektoren ϕ_i eine Orthonormalbasis von $\mathbb{R}^D$ bilden. Dann ergeben sich für einen Merkmalsvektor x, dessen Reduktion $y = (y_1, \ldots, y_d)$ mit $y_i = \phi_i^T \cdot x$, den aus y rekonstruierten Merkmalsvektor $\tilde{x} \in \mathbb{R}^D$ und den bei der Reduktion auftretenden Repräsentationsfehler $\epsilon(\tilde{x})$ die Darstellungen

$$\begin{aligned} x &= \sum_{i=1}^{D} y_i \phi_i \\ \tilde{x} &= \sum_{i=1}^{d} y_i \phi_i \qquad (2.31) \\ \epsilon(\tilde{x}) &= ||x - \tilde{x}||^2 = ||\sum_{i=d+1}^{D} y_i \phi_i||^2 \qquad (2.32) \end{aligned}$$

Ziel ist es nun, Φ für ein $d \ll D$ derart zu bestimmen, daß die Summe aller Repräsentationsfehler

$$\sum_{n=1}^{N} \epsilon(\tilde{x}_n) \qquad (2.33)$$

minimiert wird. Statt Ausdruck (2.33) wird im folgenden der Erwartungswert des Repräsentationsfehlers $\epsilon_d := E(\epsilon(\tilde{x}))$ betrachtet und dieser minimiert.

$$\epsilon_d = E(\epsilon(\tilde{x})) = E\left(||x - \tilde{x}||^2\right) = E\left(||\sum_{i=d+1}^{D} y_i \phi_i||^2\right) = \sum_{d=1}^{D} \phi_i^T \underbrace{E(x\, x^T)}_{=:S} \phi_i \qquad (2.34)$$

Die Matrix S kann aus den Trainingsdaten $\{x_n\}, n = 1, \ldots N$ geschätzt werden, und zwar als die empirische Kovarianzmatrix (siehe hierzu auch die Erläuterungen zur Maximum-Likelihood-Schätzung in Abschnitt 2.4.1):

$$\hat{S} = \frac{1}{N} \sum_{n=1}^{N} (x_n - \mu)(x_n - \mu)^T, \text{ wobei } \mu = \frac{1}{N} \sum_{n=1}^{N} x_n \qquad (2.35)$$

Nach Bestimmung der Eigenwerte $\lambda_1, \ldots, \lambda_D$ von S (bzw. von Schätzer $\hat{S}$) ergibt sich für S die Darstellung

$$S = \Phi^T \Lambda \Phi, \quad \Lambda = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_D \end{pmatrix} \qquad (2.36)$$

Das Einsetzen in Gleichung (2.34) ergibt für den Rekonstruktionsfehler

$$\epsilon_d = \sum_{i=d+1}^{D} \phi_i^T S \phi_i = \sum_{i=d+1}^{D} \phi_i^T \Phi^T \Lambda \Phi \phi_i \overset{\Phi\ Orthonormalsystem}{=} \sum_{i=d+1}^{D} e_i^T S e_i = \sum_{i=d+1}^{D} \lambda_i \qquad (2.37)$$

wobei e_i der i-te Basisvektor der kanonischen Basis von $\mathbb{R}^D$ ist. Demnach wird ϵ_d minimal, falls zur Konstruktion von $\tilde{x}$ die Eigenvektoren zu den d größten Eigenwerten von S (bzw. Schätzer $\hat{S}$) verwendet werden, d.h. es findet maximale Varianzerhaltung statt. Die durch die Transformation

$y = \Phi x$ bewirkte Projektion auf die Achsen entlang der Varianzen ϕ_i bewirkt ferner, daß die Komponenten der resultierenden Merkmalsvektoren $y = (y_1, \ldots, y_D)$ bzw. $\tilde{y} = (y_1, \ldots, y_d)$ unkorreliert sind, d.h. die Kovarianzmatrix Σ der transformierten Daten ist diagonal.

Erinnert man sich an die Motivation zur Merkmalsreduktion, nämlich die Entschärfung des Schätzproblems für die Verteilungsparameter, so hat sich das Problem in gewissem Sinne verschoben, denn auch für die KLT ist eine Kovarianzmatrix $\hat{S}$ zu schätzen. Allerdings ist dies eine Matrix, die aus allen Trainingsdaten geschätzt werden kann, während die dichtenspezifischen Kovarianzmatrizen für hohe Dichteanzahlen aus nur sehr wenigen Merkmalsvektoren geschätzt werden müssen.

Zu beachten ist, daß an keiner Stelle der KLT die Klassenzugehörigkeit der Merkmalsvektoren $x_n \in I\!R^D$ eingeht, und somit auch keine Aussagen über die Klassentrennbarkeit der transformierten Daten möglich sind. Das Beispiel in Abbildung (2.4) soll die Problematik verdeutlichen.

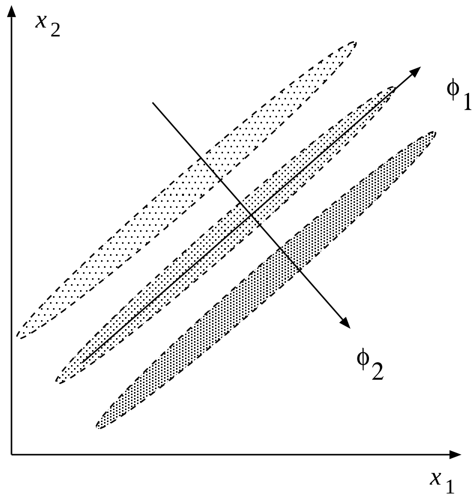

Abbildung 2.4: Da die KLT die Klasseninformation nicht berücksichtigt, sind keine in Bezug auf Klassentrennbarkeit optimalen Ergebnisse zu erwarten. In diesem Beispiel geht das diskriminative Merkmal ϕ_2 (orthogonal zur Richtung maximaler Varianz ϕ_1) bei Reduktion auf eine Dimension durch die KLT verloren [34].

2.5.2 Die Lineare Diskriminanzanalyse (LDA)

Während die im vorherigen Abschnitt dargestellte KLT die Klassenzugehörigkeit nicht berücksichtigt (und dadurch Probleme wie das in Abbildung 2.4 dargestellte auftreten können), ist das Ziel der Linearen Diskriminanzanalyse, eine lineare Transformation derart zu bestimmen, daß die mit ihr reduzierten Merkmalsvektoren möglichst gut separierbar sind, d.h. die Streuung der Daten innerhalb einer Klasse soll möglichst gering sein, während die Klassenzentren möglichst weit auseinandergezogen werden. Die folgende Darstellung ist [11] entnommen.

Zunächst sind zur Formulierung des Problems geeignete Größen zu bestimmen. Betrachtet wird die Zerlegung der empirischen Kovarianzmatrix Σ aller Trainingsvektoren, d.h. zunächst ohne Klasseninformation. Die Trainingsdaten seien hierbei dargestellt als $\{x_{nk}\}$, $n = 1, \ldots, N_k$, $k = 1, \ldots, K$.

$$\begin{aligned}
\Sigma &= \frac{1}{N} \sum_{x_{nk}} (x_{nk} - \mu)(x_{nk} - \mu)^T \\
&= \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N_k} (x_{nk} - \mu_k + \mu_k - \mu)(x_{nk} - \mu_k + \mu_k - \mu)^T \\
&= \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N_k} (x_{nk} - \mu_k)(x_{nk} - \mu_k)^T + \frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N_k} (\mu_k - \mu)(\mu_k - \mu)^T \\
&= \underbrace{\frac{1}{N} \sum_{k=1}^{K} \sum_{n=1}^{N_k} (x_{nk} - \mu_k)(x_{nk} - \mu_k)^T}_{=:S_w} + \underbrace{\frac{1}{N} \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T}_{=:S_b}
\end{aligned} \tag{2.38}$$

S_w wird *within-class-scatter*-Matrix oder Intraklassen-Streuungsmatrix, S_b *between-class-scatter*-Matrix oder Interklassen-Streuungsmatrix genannt. Nun kann kann die Wirkungsweise einer linearen Transformation W auf S_w und S_b betrachtet werden.

$$W : I\!R^D \to I\!R^d; \quad S_w \overset{W}{\mapsto} W^T S_w W; \quad S_b \overset{W}{\mapsto} W^T S_b W \tag{2.39}$$

Ziel ist nun ein möglichst großes Verhältnis der Streuung zwischen den Klassenzentren zur Streuung innerhalb jeder Klasse - mit Hilfe von W:

$$\text{maximiere } J(W) = \frac{|W^T S_b W|}{|W^T S_w W|} \tag{2.40}$$

wobei $|.|$ die Determinante einer Matrix bezeichnet. Obiges Optimierungsproblem läßt sich umformen auf das Problem, die generalisierten Eigenvektoren w_i (Spalten der Matrix W) zu den größten Eigenwerten λ_i der Gleichung

$$S_b w_i = \lambda_i S_w w_i \quad \Leftrightarrow \quad [S_b - \lambda_i S_w]\, w_i = 0 \tag{2.41}$$

zu finden. Diese werden als Nullstellen des charakteristischen Polynoms

$$\det(S_b - \lambda S_w) \overset{!}{=} 0 \tag{2.42}$$

und die Eigenvektorgleichung (2.41) für Eigenwerte λ_i und zugehörige Eigenvektoren w_i bestimmt.

Es ist zu beachten, daß S_b durch die Summe von K äußeren Produkten von Vektoren $(\mu_k - \mu)$ mit sich selbst entsteht. Die Menge aller dieser Vektoren ist linear abhängig, da $\frac{N_k}{N}(\mu_k - \mu) = -\sum_{k' \neq k} \frac{N_{k'}}{N}(\mu_{k'} - \mu)$ gilt. Somit ist der Rang von S_b maximal $K-1$, d.h. es sind auch nur maximal $K-1$ Eigenvektoren ungleich Null. Somit führt die LDA auf eine Zieldimension $d \leq K-1$.
Ebenso wie bei der KLT ergibt sich auch hier selbst ein Schätzproblem, und zwar für S_w und S_b - allerdings gelten die gleichen einschränkenden Argumente wie bei der KLT, da S_w und S_b aus allen Trainingsdaten geschätzt werden können.

2.5.3 Alternative Berechnung der LDA: Die Subspace-Methode

Die Lösung des generalisierten Eigenwertproblems (2.41) gestaltet sich manchmal als numerisch schwierig. DUDA und HART geben in [11, S. 120] eine weitere Methode an, die LDA auszuführen. Falls die within-class-scatter-Matrix S_w die Einheitsmatrix I ist, sind die Eigenvektoren in Gleichungen (2.41) die Eigenvektoren der Matrix S_b und ein Erzeugendensystem des von $(\mu_k - \mu)$, $k = 1, \ldots, K$ aufgespannten Raums. Somit ergibt sich ein dazu äquivalentes Erzeugendensystem durch Orthonormalisierung der Vektoren $(\mu_k - \mu), k = 1, \ldots, K$. Dies wird mittels Singulärwertzerlegung durchgeführt, da dies numerisch günstiger als das GRAM-SCHMIDT-Verfahren ist [29, Kapitel 2].

Im folgenden wird dieses Vorgehen als *Subspace-Methode* bezeichnet.

Die Whitening-Transformation

Die für die Durchführung der Subspace-Methode notwendige Voraussetzung $S_w = I$ wird mit Hilfe der *Whitening-Transformation* hergestellt. Hierzu wird S_w einer Hauptachsentransformation unterworfen (siehe Abschnitt 2.5.1, ohne Dimensionsreduktion) und nach der Diagonalisierung mittels Normierung der Eigenvektoren ϕ_i mit dem Faktor $1/\sqrt{\lambda_i}$, wobei λ_i der zu ϕ_i gehörende Eigenwert ist, in die Einheitsmatrix I transformiert. Für die mit der Whitening-Transformation transformierten Trainingsdaten $\{x_{nk};\, n = 1, \ldots, N_k,\, k = 1, \ldots K\}$ gilt demnach

$$S_w = \frac{1}{N} \sum_{x_{nk}} (x_{nk} - \mu_k)(x_{nk} - \mu_k)^T = I$$

2.5.4 Flexible Wahl der Zieldimension der LDA

Mit der LDA kann eine Merkmalsreduktion auf maximal $K - 1$ Merkmale durchgeführt werden. Eine geringere Zieldimension kann dadurch erzeugt werden, indem analog zum Vorgehen bei der KLT nur die größten $1 \leq d \leq K - 1$ Eigenvektoren der Lösung von (2.41) zur Konstruktion der reduzierten Merkmalsvektoren benutzt werden.

Erweist sich die durch die LDA vorgenommene Reduktion von D auf $d \leq K - 1$ Merkmale als zu hoch, d.h. die Erkennungsleistung ist durch zu hohen Informationsverlust zu stark herabgesetzt, kann diese Limitierung durch die Verwendung von *Pseudoklassen* umgangen werden.

Hierbei wird auf den Trainingsdaten zunächst ein an das EM-Training angelehnter Clustering-Algorithmus ausgeführt (siehe Abschnitt 2.4.1), mit dem die einzelnen Klassen feiner differenziert werden. Bei c Clustern pro Klasse ergibt dies $c \cdot K$ viele Pseudoklassen. Wird die LDA auf die in Pseudoklassen unterteilten Trainingsvektoren angewendet, kann eine Transformationsmatrix W erhalten werden, die eine Reduktion auf maximal $c \cdot K - 1$ Merkmale erlaubt. Durch Wahl von c und die Benutzung von ggf. weniger Eigenvektoren von W kann so eine variable Zieldimension für die Merkmalsvektoren erreicht werden.

2.6 Erzeugung virtueller Daten

Neben der Vereinfachung des statistischen Modells und der Reduktion der Merkmalsanzahl ist die Erzeugung virtueller Daten eine dritte Möglichkeit, die Verteilungsparameter zuverlässiger zu schätzen (siehe Seite 20).

Hierzu ist a-priori-Wissen über das Datenmaterial notwendig: Gesucht sind Transformationen $T : \mathbb{R}^D \rightarrow \mathbb{R}^D$, welche die Klasseninformation eines Merkmalsvektors $x \in \mathbb{R}^D$ nicht verändern, und typische, d.h. im Datenmaterial auftretende, Variationen modellieren. Bei allgemeinen Bildern kann dies z.B. die Menge aller affinen Transformationen sein. Bei der Klassifikation von Ziffern muß beachtet werden, daß durch die Toleranz beliebiger Rotationen Sechsen und Neunen verwechselt werden, d.h. die Klassenzugehörigkeit ist in diesem Fall nicht vollständig invariant gegenüber Rotation.

2.6.1 Vervielfachung der Trainingsdaten

Die Vervielfachung der Trainingsdaten beeinflußt an zwei Stellen das Erkennungssystem: Zum einen wirkt sich diese Maßnahme positiv auf die Schätzung der Verteilungsparameter $p(x|k)$ im EM-Training aus. Aufgrund einer geringen Lernstichprobe werden gerade die Varianzen oft zu klein geschätzt [25, S.54]. Virtuelle Trainingsdaten kommen ferner der Merkmalsreduktion zugute, da durch die höhere Anzahl von Referenzen auch die Kovarianzmatrix bei der KLT bzw. between-class-scatter S_b und within-class-scatter S_w bei der LDA (und der Subspace-Methode) besser geschätzt werden können (vergleiche Abschnitt 2.7.1).

2.6.2 Vervielfachung der Testdaten

Die Vervielfachung der Testdaten geht erst bei der Entscheidung des Klassifikators ein, d.h. Merkmalsreduktion und Training der Verteilungsparameter laufen davon unbeeinflußt ab. Bei Bildung der Entscheidung

$$r : x \mapsto \{1 \dots K\} \tag{2.43}$$

werden dem Klassifikator nun statt eines Vektors x mehrere Vektoren $x_1, \dots, x_t$ präsentiert, und dessen Entscheidungen $r(x_1), \dots, r(x_t)$ zu einer finalen kombiniert. Dies soll durch eine Funktion g erfolgen, die neue Entscheidungsregel für Testdatum x (hinter dem sich die aus ihm gewonnenen Vektoren $x_1, \dots, x_t$ verbergen) soll mit $r'(x)$ bezeichnet werden:

$$x \stackrel{T}{\mapsto} \{x_1, \dots, x_t\} \tag{2.44}$$

$$r'(x) = g(r(x_1), \dots, r(x_t)) \tag{2.45}$$

Dieses Vorgehen wird in [7] unter der Bezeichnung *virtual test sample Methode* (kurz VTS) dargestellt und orientiert sich an Überlegungen aus [15], wo die Kombinationsmöglichkeiten von Entscheidungen unterschiedlicher Klassifikatoren für ein Testdatum diskutiert werden.

Die Entscheidung des Klassifikators r für die aus x mittels T hervorgegangenen Merkmalsvektoren $x_1, \ldots, x_t$ berechnet jeweils die (approximierten) Wahrscheinlichkeiten $p(k|x_i), k = 1, \ldots K$. Gemäß der BAYES'schen Entscheidungsregel sollte die Entscheidung folgendermaßen getroffen werden:

$$r'(x) = \underset{k}{\operatorname{argmax}} \ \{p(k|x_1, \ldots, x_t)\} \tag{2.46}$$

Vereinfachend wird angenommen, daß die Merkmalsvektoren $x_1, \ldots, x_t$ statistisch unabhängig beobachtet werden (was streng genommen nicht der Fall ist, da die virtuell erzeugten Daten stark korreliert sind). Ferner kann angenommen werden, daß die Wahrscheinlichkeit für die Wahl einer der Transformationen aus T gleichverteilt ist, d.h. $p(x_1) = \ldots = p(x_t) = \frac{1}{t}$.
Aus (2.46) ergibt sich mit obigen Annahmen leicht die *Produktregel*:

$$r'(x) = \underset{k}{\operatorname{argmax}} \ \left\{\prod_{i=1}^{t} p(x_i|k)\right\} \tag{2.47}$$

Informell ausgedrückt bedeutet dies, daß diejenige Klasse gewählt wird, welche die höchste Wahrscheinlichkeit aufweist, daß alle Merkmalsvektoren $x_1, \ldots, x_t$ aus ihr stammen:

$$\begin{aligned} & \prod_{i=1}^{t} p(x_i|k), \quad k = 1, \ldots, K \\ = {} & P\left((x_1 \in A_k) \wedge \ldots \wedge (x_t \in A_k)\right) \overset{Modell}{=} P(x \in A_k) \end{aligned}$$

Orientiert man sich an obiger Darstellung, so ergibt sich durch eine andere Sichtweise die *Summenregel*. Hierbei wird davon ausgegangen, daß sich die Ereignisse $(x_i \in A_k)$ gegenseitig ausschließen (dies widerspricht der bei der Produktregel angenommenen statistischen Unabhängigkeit). Dies kann dadurch gerechtfertigt werden, daß von den betrachteten Transformationen stets nur eine tatsächlich auf das Testdatum angewendet wird (ein Bild ist beispielsweise nur in eine von 8 Richtungen translatiert). Für die Summenregel ergibt sich:

$$\begin{aligned} P(x \in A_k) & \overset{Modell}{=} P\left((x_1 \in A_k) \vee \ldots \vee (x \in A_k)\right) \\ & = \frac{1}{t} \sum_{i=1}^{t} p(x_i|k), \quad k = 1, \ldots, K \\ r'(x) & = \underset{k}{\operatorname{argmax}} \ \left\{\sum_{i=1}^{t} p(x_i|k)\right\} \end{aligned} \tag{2.48}$$

In [15] wird die Summenregel unter der (sehr starken) Voraussetzung, daß sich die posterior-Wahrscheinlichkeiten nur geringfügig von den a-priori-Klassenwahrscheinlichkeiten unterscheiden, hergeleitet. Motivation ist dort die in zahlreichen Experimenten gemachte Beobachtung, daß die Produktregel recht sensitiv gegenüber vereinzelt auftretenden geringen Wahrscheinlichkeiten $p(x_i|k)$ für wenige x_i, $i = 1, \ldots, t$ ist. Die Summenregel wirkt hier glättend und schneidet bei Experimenten in [15] besser ab.
Bei *Majoritätsentscheidung* (engl. *majority vote decision*) wird diejenige Klasse ausgewählt, der durch r die meisten der Merkmalsvektoren $x_1, \ldots, x_t$ zugeordnet werden.

$$\begin{aligned}\chi_{ik} &= \begin{cases} 1 & r(x_i) = k \\ 0 & r(x_i) \neq k \end{cases} \\ r'(x) &= \underset{k}{\operatorname{argmax}} \left\{ \sum\nolimits_{i=1}^{t} \chi_{ik} \right\}\end{aligned} \tag{2.49}$$

Eine weitere mögliche Entscheidung ist, für diejenige Klasse zu entscheiden, für die einer der Merkmalsvektoren $x_1, \ldots, x_t$ die höchste Wahrscheinlichkeit aufweist. Dies soll mit *Maximum-Regel* bezeichnet werden.

$$\begin{aligned} r'(x) &= \underset{k}{\operatorname{argmax}} \left\{ \max_{i=1,\ldots,t} \{p(x_i|k)\} \right\}, \\ P(x \in A_k) &\overset{Modell}{=} \max_{i=1,\ldots,t} \{P(x_i \in A_k)\} \end{aligned} \tag{2.50}$$

In der Distanzberechnung bei Einsatz von GAUSS'schen Mischverteilungen ergibt sich die Möglichkeit, für jede Dichte den günstigsten Merkmalsvektor aus $x_1, \ldots, x_t$ auszuwählen, was im folgenden *modifizierte Distanzregel* genannt werden soll.

$$r'(x) = \underset{k}{\operatorname{argmax}} \left\{ \sum_{i=1}^{I_k} c_{ki} \max_{i=1,\ldots,t} \{\mathcal{N}(x_i|\mu_{ki}, \Sigma_{ki})\} \right\} \tag{2.51}$$

Bei den in der vorliegenden Arbeit durchgeführten Experimenten schnitten Summen- und Produktregel sowie Majoritätsentscheidung ohne signifikante Unterschiede ab, so daß sich im folgenden Ergebnisse stets auf die Summenregel beziehen.

Insgesamt gesehen ist die Vervielfachung des Datenmaterials (auf Trainings- und auf Testdatenseite) zwar - wie die erzielten Ergebnisse (siehe entsprechende Teile von Abschnitt 2.7.1) zeigen - eine wirksame Methode, um die Klassifikationsleistung zu verbessern, ist aber einigen Einschränkungen unterworfen: Die Vervielfachung kostet zusätzlichen Rechenaufwand und Speicherplatz. Ein grundlegenderes Problem aber ist, daß durch die explizite Ausführung einer klassenzugehörigkeitserhaltenden Transformation nur (wenige) Stichproben einer sehr großen Menge von Merkmalsvektoren, die durch die betrachtete Transformation entstehen, erzeugt. Es wird gehofft, daß der Klassifikator die durch Stichproben sichtbare Wirkungsweise der Transformation zu extrahieren und ‚interpolieren' lernt.

Die im folgenden Abschnitt vorgestellte Tangentendistanz versucht, Invarianz gegenüber a-priori bekannten Transformationen über das bei der Klassifikation verwendete Distanzmaß zu erreichen: Die Abstandsberechnung zweier Bilder erfolgt, indem implizit alle Transformationen auf diese angewendet werden und aus den Mengen der erzeugten Bilder das zueinander ähnlichste Paar betrachtet wird. Es stellt sich aber heraus, daß die Suche in der Menge aller aus einem Bild konstruierbaren transformierten Bilder nicht effizient durchführbar ist, weshalb mit Approximationsverfahren gearbeitet werden muß. Deshalb hat auch bei der Tangentendistanz die explizite Vervielfachung von Merkmalsvektoren zur Verbesserung der Approximation eine Existenzberechtigung (siehe Abbildung 2.8, Seite 34).

2.6.3 Tangentendistanz

Motivation bei der Konstruktion der Tangentendistanz ist, a-priori-Wissen über Variabilität der Merkmalsvektoren in das bei der Klassifikation verwendete Distanzmaß zu integrieren.
Die bei Nearest-Neighbour und GAUSS'schen Mischverteilungen verwendeten Distanzmaße - besonders der Euklidische Abstand $||x - \mu||$, geringfügiger auch der Mahalanobis-Abstand $(x-\mu)^T\Sigma^{-1}(x-\mu)$ - sind sehr sensitiv gegenüber visuell recht geringen Transformationen (beispielsweise Translation), welche die Klassenzugehörigkeit nicht ändern.
Die Variabilität wird über eine Transformation

$$t : \mathbb{R}^D \times \mathbb{R}^L \to \mathbb{R}^D,\ x \mapsto t(x,\alpha) \tag{2.52}$$

mit Parametersatz $\alpha \in \mathbb{R}^L$ modelliert, die für einen Merkmalsvektor $x \in R^D$ eine Mannigfaltigkeit

$$M_x := \left\{t(x,\alpha),\ \alpha \in \mathbb{R}^L\right\} \tag{2.53}$$

erzeugt. Bei der Distanzberechnung zwischen zwei Vektoren $x \in \mathbb{R}^D$ und $\mu \in \mathbb{R}^D$ wird nun die Distanz der zueinander nächsten Repräsentanten der jeweiligen Mannigfaltigkeit betrachtet:

$$d_M(x,\mu) := \min_{\alpha_x,\alpha_\mu} \{||t(x,\alpha_x) - t(\mu,\alpha_\mu)||\} \tag{2.54}$$

Nun ist zu klären, wie $t : \mathbb{R}^D \times \mathbb{R}^L \to \mathbb{R}^D$ sinnvoll gewählt wird, d.h. welche Transformationen für das Datenmaterial charakteristisch sind. Dies wird später motiviert. Ein bedeutend gravierenderes Problem ist die Berechnung von (2.54), da dies ein schweres nichtlineares Optimierungsproblem ist. In [35] wird daher vorgeschlagen, die nichtlinearen Mannigfaltigkeiten M_x linear zu approximieren. Dies hat neben der nun einfachen Lösung des Optimierungsproblems (2.54) den Nebeneffekt, daß das entstehende Distanzmaß zwar lokal, aber nicht mehr global invariant ist, was bei einigen Transformationen durchaus erwünscht ist, etwa bei der Rotation, die zu der schon erwähnten Verwechslungsgefahr von Sechsen und Neunen führt. Eine Lineare Approximation für M_x ergibt sich durch

$$\hat{M}_x = \left\{ x + \sum_{l=1}^{L} \alpha_l \cdot x_l\ : \alpha \in \mathbb{R}^L \right\} \tag{2.55}$$

wobei $\{x_l \in \mathbb{R}^D,\ l = 1, \ldots L\}$ ein Erzeugendensystem für die Tangentialebene von M_x im Punkt $x \in \mathbb{R}^D$ ist.

In dieser Arbeit wird lediglich die Mannigfaltigkeit einer Referenz μ bei der Distanzberechnung betrachtet. Detailliertere Betrachtungen befinden sich in [14]. Für das Distanzmaß gemäß (2.54) ergibt sich somit die Tangentendistanz

$$d_t(x,\mu) = \min_{\alpha \in \mathbb{R}^L} \left\{ ||x - (\mu + \sum_{l=1}^{L} \alpha_l \cdot \mu_l)|| \right\} \tag{2.56}$$

Eine schematische Darstellung der unterschiedlichen Distanzmaße ist in Abbildung 2.5 dargestellt.

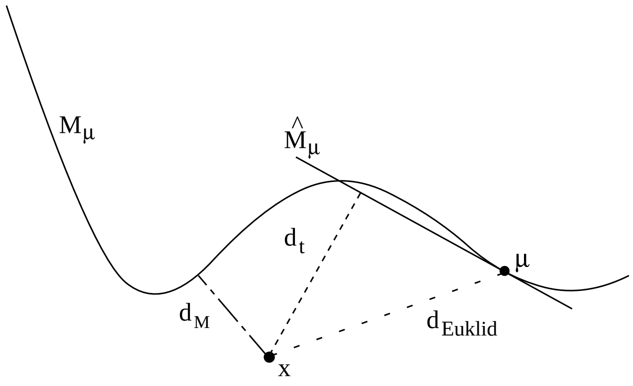

Abbildung 2.5: Schematische Darstellung von Mannigfaltigkeits-Distanz d_M (einseitig), Tangentendistanz d_t (einseitig) und Euklidischer Distanz d_{Euklid} zwischen den Merkmalsvektoren x und μ

Als Menge charakteristischer Transformationen für das Datenmaterial wird die Menge der affinen Abbildungen gewählt. Durch diese entstehende Transformationen von Koordinatenpaaren (t, s) lassen sich durch die Gleichung

$$\begin{pmatrix} t' \\ s' \end{pmatrix} = \begin{pmatrix} 1+a & b \\ c & 1+d \end{pmatrix} \begin{pmatrix} t \\ s \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix} \tag{2.57}$$

beschreiben. Im folgenden wird ein Erzeugendensystem für die Menge der affinen Abbildungen und die sich daraus ergebenden Differenzenquotienten der Grauwertfunktion I eines Bildes betrachtet, um die Tangentenvektoren zu berechnen. Dies ist [37] entnommen.

1. Horizontale Verschiebung:

$$a = b = c = d = f = 0, \quad t' = t + e, \quad s' = s$$

$$x_1(t, s) = \lim_{e \to 0} \frac{I(t+e, s) - I(t, s)}{e} \tag{2.58}$$

2. Vertikale Verschiebung:

$$a = b = c = d = e = 0, \quad t' = t, \quad s' = s + f$$

$$x_2(t, s) = \lim_{f \to 0} \frac{I(t, s+f) - I(t, s)}{f} \tag{2.59}$$

3. Rotation:

$$a = d = e = f = 0, \quad b = -c, \quad t' = t + bs, \quad s' = s - bt$$

$$\begin{aligned} x_3(t, s) &= \lim_{b \to 0} \frac{I(t+bs, s-bt) - I(t, s)}{b} \\ &= \lim_{b \to 0} \frac{I(t+bs, s-bt) - I(t, s-bt)}{b} + \lim_{b \to 0} \frac{I(t, s-bt) - I(t, s)}{b} \\ &= s x_1(t, s) - t x_2(t, s) \end{aligned} \tag{2.60}$$

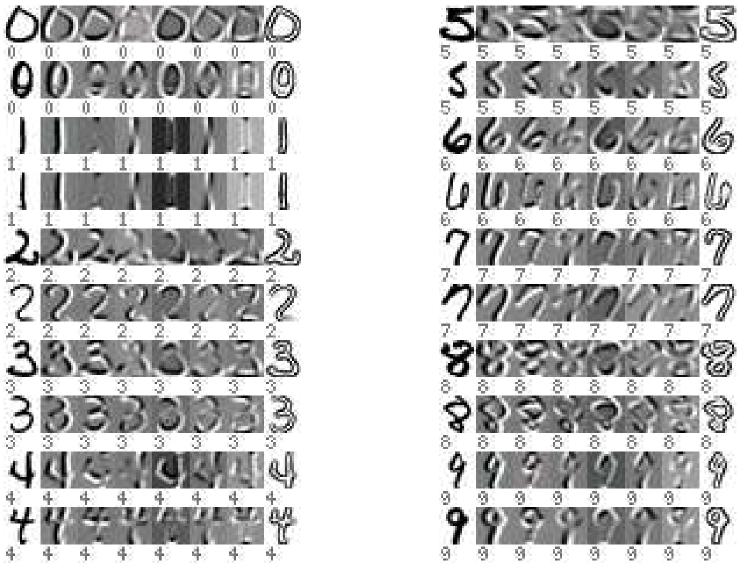

Abbildung 2.6: Visualisierung der Tangentenvektoren für einige USPS-Trainingsdaten. Bei jedem Vektor wurde dessen Wertebereich auf 0 bis 255 gespreizt.

4. Skalierung:

$$b = c = e = f = 0, \quad a = d, \quad t' = t + at, \quad s' = s + as$$

$$x_4(t,s) = \lim_{a \to 0} \frac{I(t+at, s+as) - I(t,s)}{a} = tx_1(t,s) + sx_2(t,s) \tag{2.61}$$

5. Achsendeformation:

$$a = d = e = f = 0, \quad b = c, \quad t' = t + cs, \quad s' = ct + s$$

$$x_5(t,s) = \lim_{c \to 0} \frac{I(t+cs, s+ct) - I(t,s)}{c} = sx_1(t,s) + tx_2(t,s) \tag{2.62}$$

6. Diagonale Achsendeformation:

$$b = c = e = f = 0, \quad a = -d, \quad t' = t + dt, \quad s' = s - ds$$

$$x_6(t,s) = \lim_{d \to 0} \frac{I(t+dt, s-ds) - I(t,s)}{d} = tx_1(t,s) - sx_2(t,s) \tag{2.63}$$

Die Tangentenvektoren $x_1, \ldots, x_6$ lassen sich demnach als ortsabhängige Linearkombination der Richtungsableitungen des Bildes I in x- und y-Richtung berechnen.

Nicht in den affinen Transformationen enthalten, aber für die Modellierung der Variabilität in OCR-Daten interessant, ist die Strichdicke, die als siebente Transformation hinzugenommen wird. Diese berechnet sich als der Summe der quadrierten Gradientenbilder:

7. Strichdicke:

$$x_7(t,s) = (x_1(t,s))^2 + (x_2(t,s))^2 \tag{2.64}$$

In Abbildung 2.6 sind die Tangentenvektoren für einige der Trainingsdaten des USPS-Korpus dargestellt. Die Wirkungsweise der Tangentenapproximation ist in Abbildung 2.7 dargestellt. Die Ziffern des USPS-Trainingskorpus sind gemäß Gleichung (2.55) variiert.

Abbildung 2.7: Beispiele für Variationen von USPS-Ziffern mit Hilfe der Tangentenapproximation. Links jeweils das Original, in den folgenden Spalten jeweils Variation in negative und positive Richtung der Tangenten für horizontale Translation, vertikale Translation, Rotation, Skalierung, Achsendeformation, diagonale Achsendeformation und Strichdicke. Als Vorfaktoren für die Tangenten in (2.55) wurde bei den Translationen -0.5, 0.5, bei den übrigen -0.2, 0.2 verwendet.

Zur Berechnung der Richtungsableitungen werden in dieser Arbeit die Sobel-Kantenfilter verwendet, die auf einem Grauwertbild einen gerichteten Gradienten und eine Gauß-Glättung durchführen [18, S.213]. Dies geschieht durch Anwendung folgender Filter-Templates:

$$S_x = \begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline 2 & 0 & -2 \\ \hline 1 & 0 & -1 \\ \hline\end{array} \qquad S_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline\end{array}$$

Da durch die lineare Approximation $\hat{M}_x$ Genauigkeit verlorengeht, insbesondere mit steigender Entfernung von x, kann auch hier die Trainingsdatenvervielfachung angewendet werden, um die Mannigfaltigkeit M_x besser zu approximieren. Abbildung 2.8 veranschaulicht schematisch die Wirkungsweise.

2.6.4 Schätzung tangentenähnlicher Vektoren

Die im vorherigen Abschnitt vorgestellten Tangentenvektoren modellieren zulässige Transformationen von Bilddaten, die durch a-priori-Wissen gerechtfertigt werden. In diesem Abschnitt soll untersucht werden, wie sich ‚typische' Variationen allein aus dem vorhandenen Datenmaterial ermitteln lassen. Der Ansatzpunkt hierbei ist die in Abschnitt 2.5.1 vorgestellte PCA. Wie bereits beschrieben, spiegeln die Eigenvektoren der Kovarianzmatrix Σ entsprechend der Größe ihres zugehörigen Eigenwertes Richtungen unterschiedlich großer Varianz wider. Das legt nahe, die ersten

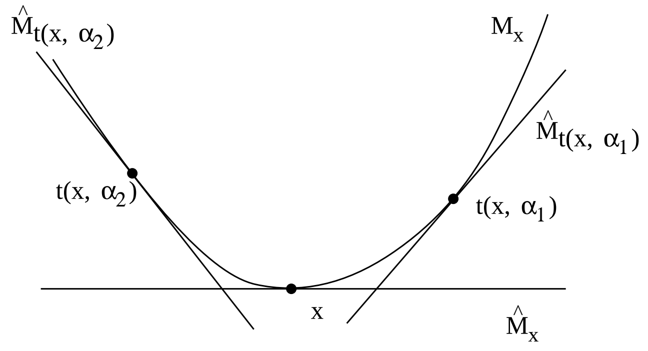

Abbildung 2.8: Die explizite Erzeugung der virtuellen Daten $t(x, \alpha_1)$ und $t(x, \alpha_2)$ zu Datum x bewirkt eine bessere Approximation der Mannigfaltigkeit M_x.

L Eigenvektoren jeder klassenspezifischen Kovarianzmatrix als klassenspezifische ‚Tangentenvektoren' zu benutzen, die auf die Mittelwertvektoren der jeweiligen Klasse wirken. Eine genaue Herleitung kann in [14, Kapitel 5.1.2] nachgelesen werden. Ähnlich wie bei der Tangentendistanz in (2.56) ergibt sich hier

$$d_t(x, \mu_k) = \min_{\alpha \in I\!R^L} \left\{ ||x - (\mu_k + \sum_{l=1}^{L} \alpha_l \cdot \mu_{kl})|| \right\}, \tag{2.65}$$

wobei $\mu_k = \frac{1}{N_k} \sum_{n=1}^{N_k} x_{nk}$ der klassenbezogene (empirische) Mittelwert ist und der Vektor μ_{kl} der Eigenvektor zu Eigenwert λ_l der klassenspezifischen (empirischen) Kovarianzmatrix Σ_k. Die Eigenwerte λ_l seien dabei absteigend sortiert, d.h. $\lambda_1 \geq \ldots \geq \lambda_D$.

Um feinere Differenzierungen vornehmen zu können, kann wieder mit Pseudoklassen gearbeitet werden, d.h. man erhält einen Prototyp-Vektor pro Cluster und berechnet die Eigenvektoren jeweils aus der clusterspezifischen Kovarianzmatrix. In Abbildung 2.9 sind die aus den clusterspezifischen Kovarianzmatrizen berechneten Vektoren für 20 Pseudoklassen auf dem USPS-Trainingskorpus dargestellt.

Interessantenweise stellt die Motivation dieses Ansatzes das Gegenteil derjenigen bei der Karhunen-Loève-Transformation dar, falls man lediglich eine Klasse betrachtet. Auftretende Distanzen in Richtung der dominierenden Variabilität werden bei Distanzmaß (2.65) vernachlässigt, statt dessen mißt man den Abstand eines Vektors von der Mannigfaltigkeit ‚typischer' - weil häufig auftretender - Transformationen. Die Distanzberechnung für KL-transformierte Daten mißt den Abstand in der Mannigfaltigkeit, also den orthogonalen Anteil. Abbildung 2.10 stellt diese Beobachtung schematisch dar. In [21] werden die Distanzmaße mit DIFS (distance in feature space) und DFFS (distance from feature space) bezeichnet.

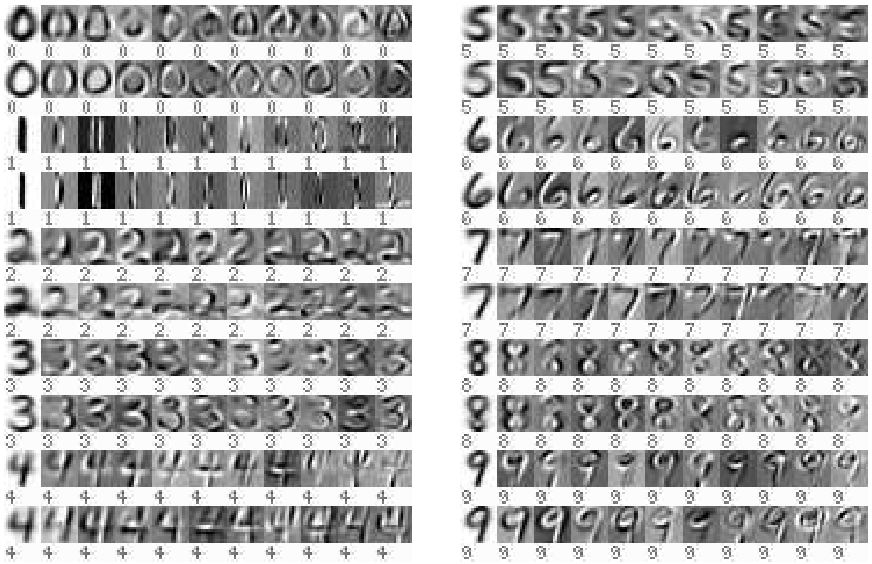

Abbildung 2.9: Aus den klassenspezifischen Kovarianzmatrizen berechnete Eigenvektoren für 20 Pseudoklassen auf dem USPS-Trainingskorpus. Ganz links jeweils der Mittelwert der Pseudoklasse.

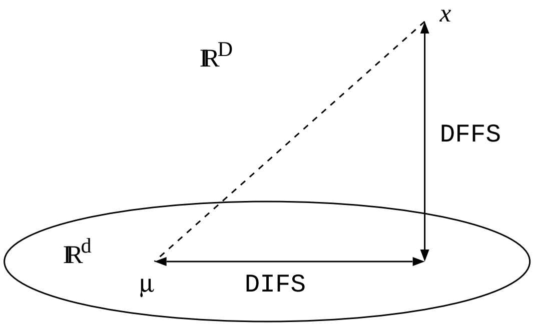

Abbildung 2.10: Vergleich von tangentenähnlichem Distanzmaß und der Distanz KL-reduzierter Daten. Die Distanz im reduzierten Merkmalsraum (DIFS) und die Distanz zum reduzierten Merkmalsraum (DFFS) zerlegen den Abstand von x und μ in die Längen zweier zueinander orthogonaler Vektoren. Vergleiche auch [21].

2.7 Ergebnisse und Diskussion

Zur Untermauerung der bisher dargestellten Betrachtungen wurden auf dem USPS-Korpus verschiedene Experimente durchgeführt, die im folgenden dargestellt und kommentiert werden sollen.

2.7.1 Durchgeführte Experimente

Zusätzlich zu den in verschiedenen Quellen angegebenen Resultaten wurden zu Vergleichszwecken eigene Versuche durchgeführt. Danach werden die mit dem auf der Bayes'schen Entscheidungsregel und Gauss'schen Mischverteilungen basierenden statistischen Klassifikator erzielten Resultate vorgestellt.

Vergleichsergebnisse aus eigenen Experimenten

Zu Vergleichszwecken wurden folgende Klassifikatoren implementiert und getestet:

1. Nearest Neighbour mit Euklidischem Distanzmaß (siehe Abschnitt 2.4.3). Die erzielten Ergebnisse sind in Tabelle 2.2 dargestellt.

2. Kernel-Densities-basierter Klassifikator (siehe Abschnitt 2.4.2). Die erzielten Ergebnisse sind in Tabelle 2.3 dargestellt.

3. Tangentendistanz-basierter Nearest Neighbour für a-priori Tangenten (siehe Abschnitt 2.6.3, Approximation der Mannigfaltigkeit nur auf Seite der Referenzen). Die erzielten Ergebnisse sind in Tabelle 2.4 dargestellt.

4. Tangentendistanz-basierter Nearest Neighbour für aus klassenspezifischen Kovarianzmatrizen geschätzte Tangentenvektoren (siehe Abschnitt 2.6.4)

Tabelle 2.2: Fehlerrate des Nearest-Neighbour-Klassifikators (Euklidische Distanz) auf USPS in Abhänigigkeit von Trainings- und Testdatenvervielfachung, keine Dimensionsreduktion.

		Testdaten	
		einfach	neunfach
Trainingsdaten	einfach (16x16)	5.6	4.7
	neunfach (18x18)	4.6	4.3

Tabelle 2.3: Fehlerrate des Kernel-Densities-Klassifikators (globale Varianz) auf USPS in Abhänigigkeit von Trainings- und Testdatenvervielfachung, keine Dimensionsreduktion

		Testdaten	
		einfach	neunfach
Trainingsdaten	einfach (16x16)	5.5	4.5
	neunfach (18x18)	4.5	4.2

Tabelle 2.4: Fehlerraten des Nearest-Neighbour-Klassifikators (Tangentendistanz) auf USPS in Abhängigkeit der Trainingsdatenvervielfachung. Die Berechnung der Tangentenmannigfaltigkeit findet nur auf Seite der Referenzen statt. Ergebnisse für einfache Testdaten.

		Fehlerrate
Trainingsdaten	einfach (16x16)	3.9
	fünffach (18x18)	3.6

Tabelle 2.5: Fehlerraten des Nearest-Neighbour-Klassifikators (DFFS) auf USPS in Abhängigkeit der Anzahl der Cluster der Trainingsdaten. Ergebnisse für einfache Testdaten.

Trainingsdaten	Fehlerrate			
	orig	20	40	80
einfach (16x16)	5.3	4.5	4.4	4.5
fünffach (18x18)	5.9	5.2	4.4	3.8
neunfach (18x18)	5.8	5.0	4.3	4.0

Diese Vergleichsergebnisse sollen helfen, die mit dem statistischen Klassifikator, basierend auf der BAYES'schen Entscheidungsregel und GAUSS'schen Mischverteilungen, erzielten Resultate einzuordnen. Diese Ergebnisse werden im folgenden dargestellt und diskutiert.

Anmerkung zur Beschränkung auf Varianzen

Der Übergang vom Schätzen voller Kovarianzmatrizen Σ_{ki} zum Schätzen von Varianzen σ_{ki}^2 (den Diagonalelementen von Σ_{ki}) stellt den ersten Schritt der Vereinfachung des Modells für $p(x|k)$ dar (siehe Abschnitt 2.4.1).
Bei der Implementierung des statistischen Klassifikators wurde auf die Benutzung von vollen Kovarianzmatrizen generell verzichtet. Wie in Abschnitt 2.4.1 bereits dargestellt, erfordert die Schätzung einer vollen, *invertierbaren* Kovarianzmatrix Σ_{ki} der Dimension $D \times D$ mindestens D, in der Praxis besser $10 \cdot D$ bis $100 \cdot D$ viele Beobachtungen, wodurch aus dem USPS-Trainingskorpus viel zu wenige Dichten schätzbar wären. Für das volle statistische Modell liegen deswegen keine Ergebnisse vor, im Zuge weitergehender Untersuchungen soll dies jedoch nachgeholt werden, insbesondere für merkmalsreduzierte Daten.

Wirksamkeit der Merkmalsreduktion

Durch die Merkmalsreduktion (siehe Abschnitt 2.5) ergibt sich ein Trade-off zwischen Informationsverlust (und damit Verringerung der potentiellen Erkennungsleistung) einerseits und der Verbesserung der Parameterschätzung für das statistische Modell der klassenbedingten Wahrscheinlichkeiten $p(x|k)$ andererseits. Ziel der Experimente ist es, den Einsatz der Merkmalsreduktion zu rechtfertigen und eine gute Abwägung zwischen beiden Effekten zu bestimmen. Es wurden folgende Experimente durchgeführt:

- Benutzung der Karhunen-Loève-Transformation (siehe Abschnitt 2.5.1)
- Benutzung der LDA (siehe Abschnitt 2.5.2, Lösung eines generalisierten Eigenwertproblems)
- Benutzung der Subspace-Methode (siehe Abschnitt 2.5.3, Lösung eines Eigenwertproblems)

In den durchgeführten Experimenten erzielten LDA und Subspace-Methode nahezu identische Ergebnisse, weshalb im folgenden stets nur die Ergebnisse für LDA-reduzierte Merkmalsvektoren angegeben ist. Tabelle 2.6 gibt eine Übersicht der erzielten Fehlerraten. Auf die Trainingsdatenvervielfachung wird in einem eigenen Abschnitt eingegangen.

Tabelle 2.6: Merkmalsreduktion auf USPS: Erzielte Fehlerraten. Ergebnisse für einfache Testdaten, globales Varianzpooling.

	Trainingsdaten	
Merkmalsreduktion	einfach (16x16)	neunfach (18x18)
keine	8.2	7.0
KLT, 39 Merkmale	5.6	3.9
LDA, 39 Merkmale	6.9	4.3

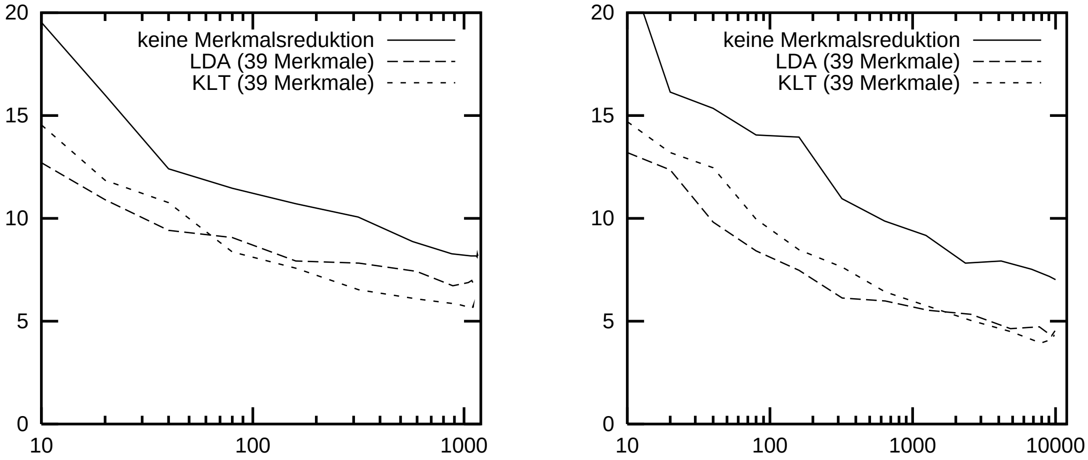

Abbildung 2.11: Merkmalsreduktion auf USPS: Erzielte Fehlerraten für einfache Trainingsdaten (links, 16x16) und verneunfachte Trainingsdaten (rechts, 18x18) in Abhängigkeit der Gesamtanzahl der Dichten. Ergebnisse für einfache Testdaten, globales Varianzpooling.

Interessanterweise erweist sich die KLT in diesem Fall gegenüber der LDA als günstiger. In [12] werden einige Eigenschaften der LDA diskutiert, u.a. daß die Optimalität im Sinne der BAYES'schen Entscheidungsregel für minimale Fehlerrate nur gilt unter der Voraussetzung, daß die Klassen jeweils GAUSS-verteilt sind und alle Klassen die gleiche Kovarianzmatrix besitzen, was sehr unwahrscheinlich ist. Ferner zeigt sich, daß beim Einsatz von VTS für LDA-reduzierte Daten bessere Ergebnisse erzielt werden (siehe Tabelle 2.9).

Es soll darüber hinaus festgestellt werden, welche Merkmalsanzahl die besten Klassifikationsleistungen erlaubt. Um eine Zieldimension d zu erreichen, wurden bei LDA und Subspace-Methode zunächst Pseudoklassen generiert (falls $d \geq K$). Da der für diese Arbeit implementierte Clustering-Algorithmus lediglich $K \cdot 2^c$, $c = 1, 2, \ldots$ Pseudoklassen liefert, wurde ggf. mit Hilfe einer nachgeschalteten LDA eine Reduktion von $K \cdot 2^c$ auf $K \leq d < K \cdot 2^c$ Merkmale vorgenommen. Die erzielten Fehlerraten sind in Tabelle 2.12 dargestellt. Bei der KLT werden für 39 Merkmale die besten Resultate erzielt (3.9 % Fehlerrate), bei der LDA ist die günstigste Zieldimension 34 (4.1%

Fehlerrate). Die Abweichung von der Fehlerrate für 39 Merkmale (4.3% Fehlerrate) ist aber gering genug, um im folgenden stets auf 39 Merkmale reduzierte Daten zu betrachten. Dies entspricht der Verwendung von 40 Pseudoklassen zur Schätzung der LDA.

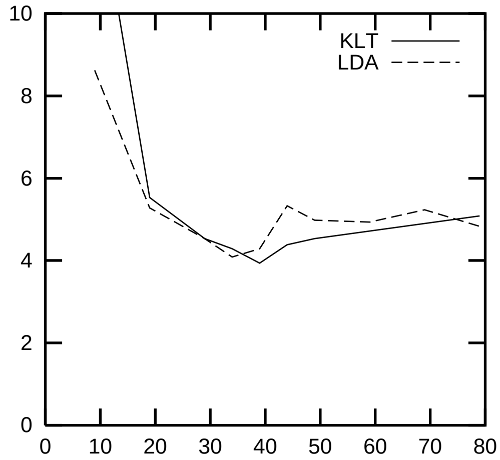

Abbildung 2.12: Merkmalsreduktion auf USPS: Erzielte Fehlerraten für verneunfachte Trainingsdaten (18x18) in Abhängigkeit der Zieldimension d. Ergebnisse für einfache Testdaten, globales Varianzpooling.

Wirksamkeit der Modellvereinfachung: Varianzpooling

Das 'Poolen' von Varianzen, d.h. der Übergang

$$\sigma_{ki}^2 \quad \rightarrow \quad \sigma_k^2 \quad \rightarrow \quad \sigma^2$$

ist eine wichtige Maßnahme, da der USPS-Trainingskorpus relativ klein ist und eine geringe Stichprobe die Gefahr birgt, die Varianz zu unterschätzen [25, S.54]. Ziel ist es, eine möglichst hohe Anzahl von Dichten in der Mischverteilung zu erreichen, um die Verteilung $p(x|k)$ genauer approximieren zu können.

Tabelle 2.7 zeigt die erzielten Ergebnisse. Deutlich erkennbar erzielt globales Pooling trotz der damit verbundenen Vereinfachung des Modells die besten Ergebnisse. Im folgenden beziehen sich Ergebnisse daher - so nicht anders angegeben - immer auf ein Modell mit globalem Varianzpooling.

Wirksamkeit der Trainingsdatenvervielfachung

Die Trainingsdatenvervielfachung (siehe Abschnitt 2.6.1) wirkt an mehreren Stellen auf das Erkennungssystem:

- In der Merkmalsreduktion sind die Matrizen Σ (für die KLT), S_w und S_b (für die LDA) bzw. S_w (für die Subspace-Methode) besser schätzbar.
- Beim EM-Training der Mischverteilungsparameter können mehr Dichten geschätzt werden. Ebenso sind die Varianzen besser schätzbar.

Tabelle 2.7: Varianzpooling auf USPS: Erzielte Fehlerraten. Die Daten sind jeweils mittels LDA auf 39 Merkmale reduziert. Ergebnisse für einfache Testdaten.

	Trainingsdaten	
Varianzpooling	einfach (16x16)	neunfach (18x18)
keins	8.5	5.9
klassenspezifisch	8.3	6.2
global	6.9	4.3

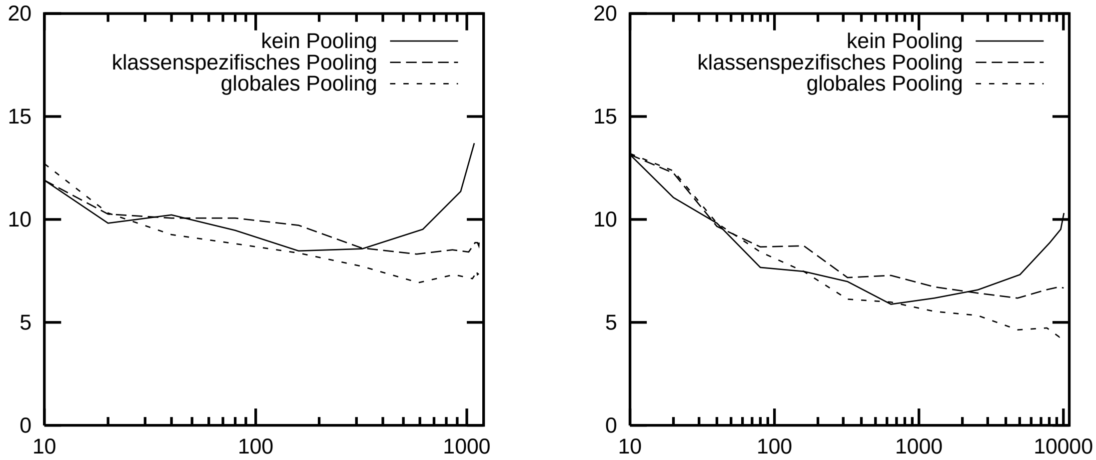

Abbildung 2.13: Varianzpooling auf USPS: Erzielte Fehlerraten für einfache Trainingsdatan (links, 16x16) und verneunfachte Trainingsdaten (rechts, 18x18) in Abhängigkeit der Gesamtanzahl der Dichten. Die Daten sind jeweils mittels LDA auf 39 Merkmale reduziert. Ergebnisse für einfache Testdaten.

- Die Erkennung wird zu gewissem Grade invariant gegenüber den bei der Datenvervielfachung verwendeten Transformationen.

Die Experimente in diesem Abschnitt sollen zeigen, daß die Vervielfachung der Trainingsdaten eine wirkungsvolle Maßnahme ist. Als Transformation zur Vervielfachung der Daten wurden Translation und Variation der Strichdicke ausgewählt, da sich diese als für den USPS-Korpus vorherrschende Variabilitäten herausgestellt haben [14, Abschnitt 7.1.3 und S. 99]. Dazu wurden folgende Experimente durchgeführt:

1. Vervielfache die Trainingsdaten durch Translation in verschieden großen Nachbarschaften

2. Vervielfache die Trainingsdaten durch eine Kombination von Translation und Strichdickenvariation

Ergebnisse für per Translation verneunfachte Trainingsdaten befinden sich in Tabelle 2.6.
Die Variation der Strichdicke kann durch einen in [18, Abschnitt 7.5.5] dargestellten Skelettierungsalgorithmus erfolgen. Dieser ist allerdings für Binärbilder definiert. Auf die Einschränkung

der Klassifikationsleistung beim Übergang von Grauwertbildern zu Binärbildern wird in [14, Abschnitt 7.1.4] eingegangen. Deshalb wurde als alternative Implementierung die Modifikation eines Merkmalsvektors durch Addition bzw. Subtraktion der Strichdicken-Tangente (siehe Abschnitt 2.6.3) gewählt.

Eine Kombination von Translation und Strichdickenvariation (Faktor 9 für Translation, Faktor 3 für Strichdicke, also insgesamt Vervielfachung um den Faktor 27) ergab keine Verbesserung der Ergebnisse, die in Tabelle 2.6 (Verneunfachung der Trainingsdaten durch Translation) dargestellt sind. Versuche mit per geringfügiger Rotation vervielfachten Trainingsdaten verliefen ebenfalls erfolglos.

Ferner soll untersucht werden, an welchen Stellen des Erkennungssystems die resultierenden Gewinne erzielt werden, d.h. inwieweit die Verbesserungen auf die zuverlässigere Schätzung der within-class-scatter-Matrix S_w bei der Subspace-Methode (siehe Abschnitt 2.5.3) oder durch die zuverlässigere Schätzung der Verteilungsparameter im EM-Training (siehe Abschnitt 2.4.1) zurückzuführen sind. Dazu wurde folgendes Vorgehen gewählt:

- Führe die Schätzungen im Rahmen der Merkmalsreduktion mit Hilfe vervielfachter Trainingsdaten aus, das Training der Mischverteilungsparameter erfolgt lediglich anhand der normalen Testdaten.
- Der umgekehrte Fall: Benutze die normalen Trainingsdaten zur Berechnung der Merkmalsreduktion, schätze die Mischverteilungsparameter über die vervielfachten Trainingsdaten.

Tabelle 2.8: Trainingsdatenvervielfachung auf USPS: Einfluß auf die Berechnung der Merkmalsreduktion und die Parameterschätzung. Die Daten sind jeweils mittels LDA auf 39 Merkmale reduziert. Ergebnisse für einfache Testdaten.

		Training der GMV-Parameter	
		einfach (16x16)	neunfach (16x16)
Berechnung der Merkmalsextraktion	einfach (16x16)	6.9	6.3
	neunfach (16x16)	5.4	5.0
		einfach (18x18)	neunfach (18x18)
Berechnung der Merkmalsextraktion	einfach (18x18)	7.4	5.5
	neunfach (18x18)	5.9	4.3

Die Ergebnisse für diese Experimente sind in Tabelle 2.8 dargestellt. Der Einfluß der Trainingsdatenvervielfachung scheint demnach besonders die Schätzung der Merkmalsextraktion zu verbessern.

Wirksamkeit der Testdatenvervielfachung: VTS

Die Wirksamkeit der Vervielfachung von Testdaten und des Einsatzes der *virtual test sample* Methode (VTS) soll durch Experimente belegt werden. Bemerkenswert ist, daß alle vorangegangenen

Schritte des Prozesses (Merkmalsreduktion und Training der Verteilungsparameter) unverändert beibehalten werden können. Diese Maßnahme kann also - auch aufgrund der allgemein relativ geringen Kosten für die Klassifizierung eines Vektors über die Auswertung der Mischverteilungsfunktion - sehr effizient angewendet werden.

Als mögliche Transformationen bieten sich hier wieder die im vorherigen Abschnitt diskutierten Transformationen an: Translation und Strichdicke. Aufgrund der guten Ergebnisse im Zusammenhang mit der Trainingsdatenvervielfachung wurden die Experimente mit per Translation vervielfachten Testdaten durchgeführt. Die erzielten Fehlerraten befinden sich in Tabelle 2.9.

Tabelle 2.9: Testdatenvervielfachung auf USPS: Erzielte Fehlerraten.

Trainingsdaten	Testdaten	
(neunfach, 18x18)	einfach	neunfach
keine Merkmalsreduktion	7.0	6.0
KLT, 39 Merkmale	3.9	3.7
LDA, 39 Merkmale	4.3	3.4

Von den in Abschnitt 2.6.2 dargestellten Regeln erwiesen sich Maximumregel, Summenregel, Produktregel und Majoritätsentscheidung als nahezu identisch bei den erzielten Ergebnissen. Die modifizierte Distanzregel schneidet stets schlechter ab.

Interessant an den Ergebnissen in Tabelle 2.9 ist der geringe Zugewinn durch VTS bei KLT-reduzierten Daten. Eine mögliche Erklärung hierfür ist, daß durch die starke Korrelation der virtuellen Daten auch deren Eigenraum-Repräsentationen sehr eng zusammenliegen, was eine nur geringe Menge an Zusatzinformationen liefert, die VTS nutzen kann.

2.7.2 Konkurrierende Klassifikatormodelle

In diesem Abschnitt sollen -recht kompakt- die Ansätze konkurrierender Klassifikatormodelle vorgestellt werden, um eine Einordung der in dieser Arbeit dargestellten Methoden zu erlauben.

Künstliche neuronale Netze

In der vorliegenden Arbeit wird durch die Entscheidungsfunktion r diejenige Klasse ausgewählt, welche die maximale klassenbedingte Wahrscheinlichkeit für einen zu klassifizierenden Merkmalsvektor $x \in \mathbb{R}^D$ besitzt, d.h. r entscheidet nach Auswertung eines statistischen Modells. Prinzipiell sind jedoch Antworten beliebiger *Diskriminantenfunktionen* $g(x,k)$ denkbar:

$$r : \mathbb{R}^D \rightarrow \{1,\ldots,K\}; \quad r(x) = \underset{k}{\operatorname{argmax}} \ \{g(x,k)\}$$

Ein Beispiel sind *polynomielle Klassifikatoren*, die für einen Merkmalsvektor $x = (x_1,\ldots,x_D) \in \mathbb{R}^D$ einen Antwortwert berechnen, der sich als gewichtete Summe von Potenzen in x_i, $i = 1,\ldots,D$ berechnet, z.B. quadratisch: $g(x,k) = \alpha_{k0} + \sum_{d=1}^{D} \alpha_{kd} \cdot x_d + \sum_{d=1}^{D} \sum_{e=1}^{E} \beta_{kde} \cdot x_d x_e$

Ein Neuronales Netz stellt eine weitere Möglichkeit dar, eine Diskriminantenfunktion zu berechnen (siehe [25], S. 63ff).
Eine Einführung zu Neuronalen Netzen befindet sich etwa in [23]. An dieser Stelle soll lediglich das *Multilayer-Perceptron* vorgestellt werden. Es besteht aus in mehreren Schichten angeordneten Knoten, die in Anlehnung an die nachgebildete Realität – das menschliche Gehirn – auch *Neuronen* genannt werden. Jedes Neuron arbeitet parallel und unabhängig von allen anderen. Verbindungen zwischen Neuronen existieren beim Multilayer-Perceptron von Schicht zu Schicht. Die Eingabeschicht enthält für jede Komponente des Merkmalsvektors ein Eingabeneuron, die Ausgabeschicht pro Klasse ein Neuron, an dem – nach Eingabe eines Merkmalsvektors in das Netz – der Wert der Diskriminantenfunktion $g(x,k)$ für die jeweilige Klasse anliegt. In jedem Neuron werden die Werte der Eingänge $y_1,\dots,y_L$ gewichtet summiert und anschliessend in die sogenannte Aktivierungsfunktion f des Neurons gegeben. Der berechnete Wert wird schließlich an nachfolgende Neuronen weitergegeben. Prinzipiell kann f für jedes Neuron separat gewählt werden, meist verwendet man jedoch eine einzige Funktion. Als Wertebereich der Aktivierungsfunktion f wird in der Regel $\{0,1\}$ oder $\{-1,1\}$ gewählt, für f selbst beispielsweise die Sigmoid-Funktion $f(x)=\frac{1}{1+e^{-x}}$. Deren Nichtlinearität ist eine wichtige Eigenschaft, da das Netz für lineare f lediglich in der Eingabe lineare Funktionen generieren kann.

$$y_i = f\left(\alpha_{i0} + \sum_{l=1}^{L} \alpha_{il} \cdot y_l\right)$$

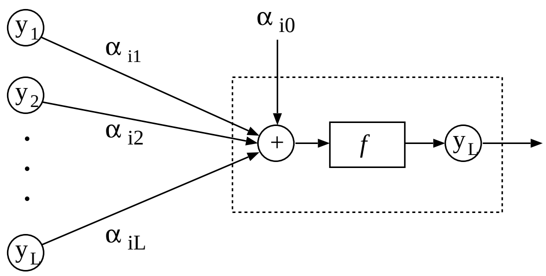

Abbildung 2.14: Schematischer Aufbau eines Neurons. Die Werte an den Eingängen $y_1,\dots,y_L$ werden mit $\alpha_{i1},\dots,\alpha_{iL}$ \$ gewichtet summiert. α_{i0} dient dazu, auch konstante Werte generieren zu können. Die gewichtete Summe wird in die Aktivierungsfunktion f eingesetzt und der resultierende Wert an die Nachfolgerknoten weitergegeben.

Die Funktionsweise eines Neuronalen Netzes kann folgendermaßen beschrieben werden [23]: In der *Ruhephase* sind alle Aktivierungen, d.h. Neuronenausgänge, konstant. Wird dem Multilayer-Perceptron ein Merkmalsvektor präsentiert, beginnt die *Arbeitsphase*, in der alle Neuronen ihre Aktivierung neu berechnen, bis das Netz wieder in eine Ruhephase gelangt. Das Lernen eines Neuronalen Netzes besteht darin, die Gewichte α_{il} derart einzustellen, daß die in der Ausgabeschicht erhaltenen Werte möglichst genau die Werte der gewünschten Diskriminantenfunktion $g(x,k)$ annehmen. Der Gesamtfehler wird im Training als die Summe der quadrierten Abweichungen von Sollwert und Istwert gemessen. Ein zum Training benutzter Algorithmus ist die *error backpropagation*. Diese führt ein Gradientenabstiegsverfahren auf der Fehlerfunktion, die in Abhängigkeit der aktuellen Wahl der α_{il} formuliert wird, aus.

In [16] wird eine Methode vorgestellt, problemspezifisches Wissen in ein Multilayer- Perceptron zu integrieren, und zwar für die Erkennung handgeschriebener Ziffern. Das dort beschriebene Multilayer-Perceptron LeNet 1 besitzt neben Eingabe- und Ausgabeschicht noch 4 innere Schichten, in denen unterschiedlich große *'feature maps'* berechnet werden, die aus abwechselnden Anwendungen von Faltung und Verkleinerung von Kartenteilbereichen entstehen. Dies ist schematisch in Abbildung 2.15 dargestellt. Per Konstruktion entstehen lokale Invarianzen der Merkmalskarten gegenüber bestimmten Transformationen, durch das Subsampling beispielsweise in Bezug auf die Translation.

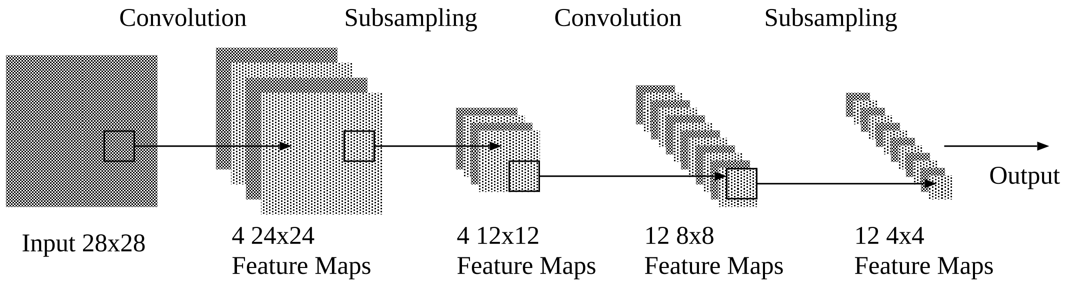

Abbildung 2.15: Aufbau von LeNet 1. Um das Berechnen der Feature Maps zu vereinfachen, wird eine 28x28 Pixel große Eingabeschicht verwendet. Das USPS-Datum (16x16 Pixel) wird zentriert eingesetzt. Die berechneten Features werden mit wachsendem Weg durch das Netz komplexer.

Durch Formulierungen von Nebenbedingungen und Abhängigkeiten für die Gewichte besitzt LeNet1 bei 4635 Knoten und 98442 Verbindungen nur 2578 freie Parameter, verglichen mit beispielsweise 123.300 freien Parametern, die ein vollständig verbundenes Netz mit 300 inneren Knoten besitzt, welches in [17] erwähnt wird.

- Neuronale Netze stellen eine Möglichkeit dar, eine Diskriminantenfunktion $g(x, k)$ aus Trainingsdaten zu erlernen. Bei der Klassifikation eines Musters x wird für diejenige Klasse k entschieden, für die $g(x, k)$ das Maximum annimmt.

- Problemspezifisches Wissen kann über die gezielte Verschaltung von Neuronen und die Formulierung von Nebenbedingungen sowie Abhängigkeiten der Gewichte α_{il} in das Neuronale Netz eingebracht werden.

- Der zumeist verwendete Trainingsalgorithmus, die *error backpropagation*, erfordert besonders sorgfältige Implementierung [25], und führt eine lokale Suche bei der Parameteroptimierung aus.

Die Support-Vektor-Maschine

Support-Vektor-Maschinen (SVMs), auch mit *optimal margin classifier* bezeichnet, arbeiten ohne statistische Modelle für die Verteilung des Datenmaterials im Merkmalsraum. Eine Einführung zu diesem Thema befindet sich beispielsweise in [3]. Ziel bei der Konstruktion einer Entscheidungsfunktion r durch eine Support-Vektor-Maschine ist es, die Generalisierungsfähigkeit von r zu maximieren. Hierzu wird als Maß das sogenannte 'Erwartete Risiko' $R(\alpha)$ für einen allgemeinen Klassifikator mit Parametersatz α benutzt. Mit Wahrscheinlichkeit $1-\eta$ gilt:

$$R(\alpha) = R_{\mathrm{emp}}(\alpha) + \sqrt{\frac{h(\log(2l/h)+1) - \log(\eta)}{l}} \tag{2.66}$$

Demnach hängt die zu erwartende Fehlerrate für r mit Parametersatz α neben der Fehlerrate R_{emp} auf den Trainingsdaten (l Stück) auch von der sogenannten VC-Dimension (von Vapnik, Chervonenskis) h der Entscheidungsfunktion r ab. Formal ist h die maximale Anzahl von Trainingsvektoren, die durch die Entscheidungsfunktion separabel sind und kann als Maß für die Komplexität der verwendeten Entscheidungsfunktion interpretiert werden, allerdings nicht im Sinne der Parameteranzahl, wie Beispiele in [3] veranschaulichen: Eine hohe Parameteranzahl bewirkt nicht zwangsläufig eine hohe VC-Dimension, ebensowenig wie eine niedrige Parameteranzahl eine niedrige VC-Dimension.

Support-Vektor-Maschinen versuchen, $R(\alpha)$ in Gleichung (2.66) zu minimieren, indem h minimiert wird. Anschaulich erscheint klar, daß eine gute Erklärung der Trainingsdaten (R_{emp} niedrig) durch eine Funktion r niedriger Komplexität (h niedrig) eine hohe Wahrscheinlichkeit besitzt, daß tatsächlich eine diskriminative Eigenschaft des Datenmaterials gefunden wurde, die konstruierte Entscheidungsfunktion r also eine gute Fähigkeit zur Generalisierung besitzt.

Allgemein besteht der Ansatz einer Support-Vektor-Maschine bei einem 2-Klassen-Problem darin, eine Hyperebene

$$H: \quad w \cdot x + b = 0 \tag{2.67}$$

mit Normalenvektor w im Merkmalsraum $\mathbb{R}^D$ zu bestimmen, welche die Trainingsdaten $\{(x_i, y_i),\ x_i \in \mathbb{R}^d, y_i \in \{-1,1\}\}$ mit *maximalem* Abstand separiert. (Um Formulierungen zu vereinfachen, wird die Menge der Klassenindizes als $\{-1,1\}$ definiert).
Die Klassifikation eines Testdatums $x \in \mathbb{R}^D$ erfolgt dann durch Bestimmung der Lage von x bzgl. H:

$$r(x) = \mathrm{sgn}(wx+b) \tag{2.68}$$

Die Separation der Trainingsdaten wird ausgedrückt durch

$$x_i \cdot w + b \geq +1 \quad ; \quad y_i = +1 \tag{2.69}$$

$$x_i \cdot w + b \leq -1 \quad ; \quad y_i = -1 \tag{2.70}$$

was sich zu einer Darstellung zusammenfassen läßt:

$$y_i(x_i \cdot w + b) \geq 1 \quad \forall i \tag{2.71}$$

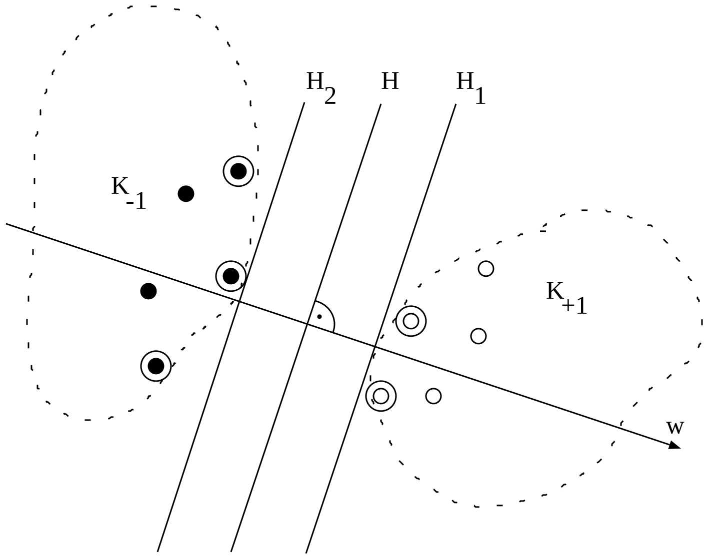

Abbildung 2.16: Separation zweier Klassen durch eine Support-Vektor-Maschine: Die SVM konstruiert diejenige Hyperebene H, welche die Daten der beiden Klassen K_{-1} und K_{+1} mit maximalem Abstand separiert. Merkmalsvektoren, die H am nächsten liegen (und somit die Lage von H bestimmen), sind markiert. Diese werden *Support-Vektoren* genannt.

Durch die Ungleichungen (2.69) und (2.70) ergeben sich 2 parallele Hyperebenen

$$\begin{aligned} H_1: \quad & w \cdot x_i - (-b+1) &= 0 \\ H_2: \quad & w \cdot x_i - (-b-1) &= 0 \end{aligned}$$

welche voneinander den Abstand

$$d = \frac{-b+1}{||w||} - \frac{-b-1}{||w||} = \frac{2}{||w||} \tag{2.72}$$

haben. Das Optimierungsproblem besteht darin, Ausdruck (2.72) unter den Nebenbedingungen (2.71) zu maximieren. Die Situation ist in Abbildung 2.16 dargestellt.

Durch die Aufteilung eines K-Klassen-Problems in K viele 2-Klassen-Probleme lassen sich auch Klassifikationsaufgaben mit $K > 2$ behandeln.

Durch Erweiterungen werden Anwendbarkeit und Leistungsfähigkeit der Support-Vektor-Maschine weiter erhöht: Für den Fall, daß die Trainingsdaten nicht fehlerfrei separierbar sind, wird in [4] ein Mechanismus vorgestellt, der die Verletzung von (2.71) in meßbarem Maße erlaubt und das Problem der Separation der Trainingsdaten mit maximalem Abstand um die simultane Minimierung der Separationsverletzung erweitert. Die Nebenbedingungen (2.71) werden modifiziert zu

$$y_i(x_i \cdot w + b) \geq 1 - \xi_i \quad \forall i \tag{2.73}$$

Schematisch ist dies in Abbildung 2.17 dargestellt.
Die zu minimierende Funktion enthält nun den mit γ gewichteten Term, der die Separationsfehler mißt:

$$\frac{1}{2}||w||^2 + \gamma \cdot \sum_i \xi_i \tag{2.74}$$

Ein zweiter Punkt ist die implizite Generierung nichtlinearer Separationsflächen durch Einsatz sogenannter *Kernfunktionen*. Wie z.B. in [32] dargestellt, lassen sich Algorithmen, die über Skalarprodukte formuliert werden können, durch die Substitution der auftretenden Skalarprodukte erweitern. Motivation dabei ist, die Daten mittels einer Abbildung Φ nichtlinear in einen höherdimensionalen Raum zu transformieren, in dem das Skalarprodukt (bezogen auf den ursprünglichen Merkmalsraum) 'mächtiger' ist. Im Fall der *Kernel-PCA* [32, Kapitel 3] bedeutet beispielsweise der Einsatz polynomieller Kernfunktionen die Berücksichtigung von Korrelationen höherer Ordnung. Um die Schwierigkeit der Berechnung des Skalarproduktes $\Phi(x_i) \cdot \Phi(x_j)$ im hochdimensionalen Raum zu umgehen, wird statt dessen der Wert einer Kernfunktion $K(x_i, x_j)$ berechnet, d.h. bei Berechnungen wird der bisherige Mermalsraum nicht verlassen, die Abbildung Φ findet lediglich implizit statt. Neben der PCA wurde dies auch erfolgreich für die Diskriminanzanalyse durchgeführt [20].
In Support-Vektor-Maschinen kann dieses Vorgehen eingebracht werden, da das Optimierungsproblem (2.74) durch die Darstellung von w als Linearkombination von Support-Vektoren entsprechend umformuliert werden kann.

Interessant sind im Vergleich zu den anderen vorgestellten Klassifikatoren folgende Punkte:

- Support-Vektor-Maschinen machen keine Annahmen über die statistische Verteilung des Datenmaterials im Merkmalsraum.

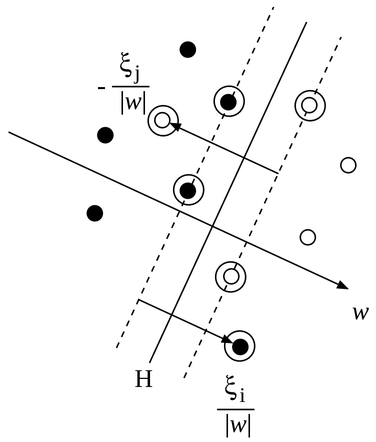

Abbildung 2.17: Erweiterung der Support-Vektor-Maschine auf nicht separierbare Trainingsdaten: Die Variablen ξ_i bilden ein Maß für die Verletzung der Separationsbedingung (2.71). Diese wird umformuliert zu neuen Nebenbedingungen (2.73)

- Support-Vektor-Maschinen minimieren das *erwartete Risiko* bei der Konstruktion einer Entscheidungsfunktion. Durch deren niedrige VC-Dimension bietet diese gute Fähigkeit zur Generalisierung. Die beim statistischen Ansatz dieser Arbeit verwendete Maximum-Likelihood-Schätzung der Verteilungsparameter hingegen bedeutet eine Minimierung des *empirischen Risikos*.

- Lediglich ein Teil der Trainingsdaten ist letztlich für die Konstruktion der Entscheidungsfunktion relevant, und zwar diejenigen Merkmalsvektoren, welche der Trennfläche am nächsten liegen. Diese werden *Support-Vektoren* genannt. Diese sind a-priori unbekannt. Aufgrund der Leere auch des reduzierten Merkmalsraums liegt der Anteil von Support-Vektoren an den Trainingsdaten auf dem USPS-Korpus bei bis zu 42% (siehe [32], Abschnitt 2.5.2).

- Support-Vektor-Maschinen arbeiten prinzipiell ohne problemspezifisches Wissen. Dieses kann über *virtuelle Support-Vektoren* eingebracht werden, was im wesentlichen Trainingsdatenvervielfachung entspricht [32, Kapitel 4].

- Für das Problem, welches zur Bestimmung der Entscheidungsfunktion zu lösen ist, kann numerisch das globale Optimum gefunden werden. Dies ist ein Unterschied in Bezug auf das lokale Optimierungsverfahren beim Neuronalen Netz.

2.7.3 Zusammenfassung

Das konstruierte Erkennungssystem, welches mit erscheinungsbasierter Merkmalsanalyse, Linearer Diskriminanzanalyse zur Merkmalsreduktion sowie einem statistischen Klassifikator auf Grundlage der BAYES'schen Entscheidungsregel und der Modellierung der Wahrscheinlichkeiten über

Tabelle 2.10: Einordung der erzielten Ergebnisse für den USPS-Datenkorpus

Methode/Publikation	Fehlerrate in %
Mensch [35]	2.5
Neuronales Netz, LeNet1 [16]	*4.5
NN, Euklidische Distanz [35]	**5.9
NN, Tangentendistanz [35]	**2.6
Support-Vektor-Maschine [32]	4.2
Support-Vektor-Maschine mit virtuellen Support-Vektoren [32]	3.2
i6:	
Mischverteilungen, Tangentendistanz, VTS [7]	2.7
Tangentendistanz, Kernel Densities, VTS, Bagging [14]	2.2
Diese Arbeit:	
GAUSS'sche Mischverteilungen, virtuelle Trainingsdaten	4.3
GAUSS'sche Mischverteilungen, virtuelle Trainingsdaten, VTS	3.4

*, **: Ergebnisse sind nur eingeschränkt vergleichbar, Erläuterungen siehe Seite 15

GAUSS'sche Mischverteilungen arbeitet, hat sich als konkurrenzfähig zu anderen Klassifikationsansätzen erwiesen.

Tabelle 2.10 stellt die erzielten Ergebnisse im Vergleich zu bisher publizierten Ergebnissen dar.
Es wurden Methoden vorgestellt, mit denen das Problem der zuverlässigen Parameterschätzung auch auf kleinen Trainingskorpora behandelt werden kann. Dies geschieht durch sinnvolle Vereinfachungen des statistischen Modells und statistisch motivierte Merkmalsreduktion.

Eine besondere Eigenschaft ist die allgemeine Verwendbarkeit des Klassifikators, was beispielsweise in [33] einging. Dort bestanden die zu klassifizierenden Merkmalsvektoren nicht aus Bildmaterial, sondern medizinischen Meßwerten. Das Erkennungssystem konnte für diese Aufgabe ohne Modifikationen eingesetzt werden. Weiterhin wurde das System erfolgreich zur Klassifikation roter Blutzellen eingesetzt [6].
A-priori-Wissen über das verwendete Datenmaterial kann an klar definierten Stellen in das Erkennungssystem eingebracht werden, was in dieser Arbeit durch die explizite Generierung virtueller Trainings- und Testdaten erfolgt. In [7] ist beschrieben, wie die Tangentenapproximation benutzt werden kann, um die Mittelwerte und Kovarianzmatrizen innerhalb GAUSS'scher Mischverteilungen besser zu schätzen und die Invarianz gegenüber klasseninformationserhaltenden Transformationen ohne explizite Datenvervielfachung auf der Trainingsseite zu erreichen.

Kapitel 3

Statistische Klassifikation für den Bildzugriff

Ein Ziel dieser Diplomarbeit ist es, die Tauglichkeit der im vorherigen Kapitel vorgestellten Methoden für den Bildzugriff zu untersuchen. Es wird versucht, das bei der Einzelobjekterkennung verwendete Verfahren, die erscheinungsbasierte Klassifikation von Bilddaten fester Größe, auf allgemeineres Bildmaterial anzuwenden. Dieses kann Objekte einer bekannten Menge enthalten, wobei allerdings neben der Klassenzugehörigkeit auch deren Lage, Skalierung und Anzahl unbekannt sind. Im folgenden Abschnitt soll auf die Unterschiede der Aufgabenstellung und die dadurch möglicherweise auftretenden Probleme eingegangen werden.

3.1 Einleitung und Stand der Technik

Der Anwendungsbereich folgender Untersuchungen ist das sogenannte *Image Retrieval*, d.h. das Indizieren von Bildmaterial, um anschließend inhaltsbasiert darauf zugreifen zu können. Das bedeutet, der Vergleich von Bildern findet nicht (zwangsläufig) auf Pixelebene statt (was unter Umständen auch sehr rechenintensiv ist), sondern anhand von in einer Datenbank gespeicherten Attributen, die das System den Bildern möglichst automatisch zuweist. Je nach Sichtweise unterscheiden sich 2 Bilder, die im System mit gleichen Attributen assoziiert werden, unterschiedlich stark, da z.B. ein in beiden Bildern vorhandenes Objekt in unterschiedlichen Lageparametern vorliegt. Der Begriff der Ähnlichkeit ist hier sehr viel stärker von der Ähnlichkeit auf Pixelebene losgelöst. Die Ansätze zur Bildindizierung allgemein sind sehr unterschiedlich. Ein kurzer Überblick zu existierenden Systemen und deren grundlegenden Techniken befindet sich etwa in [36]. Prinzipiell versucht ein System, aus einem Bild *signifikante* Merkmale zu extrahieren, anhand derer ein Testbild mit anderen Bildern im System in Relation gesetzt werden kann. Diese Merkmale können u.a. aus

- Farbinformationen
- Texturinformationen

- Forminformationen

des Bildes gewonnen werden. In [36, 9] werden Konzepte der Einzelobjekterkennung auf ganze Bilder angewendet. Ein weiterer Ansatz ist, im Bild vorhandene Objekte zu identifizieren (dies umfaßt Lokalisierung und Klassifikation) und diese Informationen als Attribute für das Bild in der Datenbank zu benutzen. Dieser Ansatz soll in der vorliegenden Arbeit weiter betrachtet werden. Ein ähnliches Ziel, nämlich die Lokalisierung und Klassifikation von Gesichtern in allgemeinem Bildmaterial, verfolgen eine Reihe von Systemen.
Eines von diesen soll wegen seiner dieser Arbeit ähnlichen Vorgehensweise vorgestellt werden. Es handelt sich dabei um die von MOGHADDAM und PENTLAND in [21] publizierte Arbeit. Dort wird ein erscheinungsbasierter Ansatz gewählt, d.h. der Algorithmus arbeitet ohne die Extraktion von Kanteninformationen, lokalen Merkmalen (z.B. Analyse der Fourier-Transformierten eines Bildes) oder ähnlichem. Zur effizienten Codierung von Gesichtsbildern hat sich die *Eigenraum*-Repräsentation als geeignet herausgestellt, d.h. eine Merkmalsreduktion der Gesichtsbilder mittels der KLT. Um ein Gesicht in einem Testbild zu finden, wird konzeptionell ein Fenster fester Größe über das Bild geschoben und dessen Inhalt jeweils in den aus Trainingsbildern generierten Eigenraum transformiert. Nun können zwei Größen betrachtet werden: Die Distanz des merkmalsreduzierten Fensterinhalts zu einer merkmalsreduzierten Referenz (DIFS, *distance in feature space*) und der bei der Transformation auftretende Fehler (DFFS, *distance from feature space*). Dieser Sachverhalt ist in Abbildung 3.1 dargestellt.

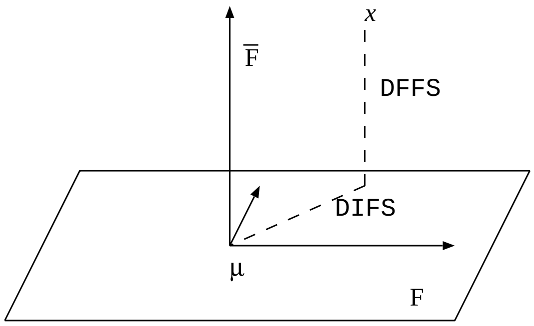

Abbildung 3.1: Aufteilung des Merkmalsraums in Eigenraum F und den dazu orthogonalen Raum $\bar{F}$ nach [21]. Die Distanz in F wird mit DIFS (distance in feature space), die Distanz in $\bar{F}$ mit DFFS (distance from feature space) bezeichnet.

Hierbei ist die Größe DFFS ein Kriterium dafür, ob sich an der aktuell betrachteten Fensterposition ein Gesicht befindet. Die Beachtung dieser Größe ist wichtig, da bei reiner Betrachtung der DIFS eine zu hohe Rate von *false alarms*, also fehlerhaft als Gesichtsposition gemeldeter Bildregionen, auftritt. Für die eigentliche Klassifikation der Gesichter (nach verschiedenen Personen) interessant ist die Distanz DIFS von Beobachtung und Referenzen im merkmalsreduzierten Raum. Die Wahrscheinlichkeit $p(x|k)$, daß der aktuell betrachtete Merkmalsvektor x ein Gesicht der Klasse k darstellt, wird modelliert als

$$p(x|k) = p_F(x|k) \cdot p_{\bar{F}}(x|k) \tag{3.1}$$

Das Abfahren des Bildes mit dem sliding window (repräsentiert durch eine Indexmenge R, $|R| = D$) erzeugt eine *Karte* $S(i, j|k)$ (in [21] werden diese *saliency maps* gennannt), die für jede Bildposition (i, j) die Wahrscheinlichkeit für das Vorhandensein eines Objekts der Klasse k in der mit (i, j) assoziierten Region $R_{(i,j)} = (I(i + r, j + c) : (r, c) \in R)$ enthält:

$$S(i, j|k) = p(x_{(i,j)}|k) \tag{3.2}$$

wobei $x_{(i,j)} \in \mathbb{R}^D$ der aus Region $R_{(i,j)}$ gewonnene Merkmalsvektor ist.

Die Lokalisierung eines Objekts (in diesem Fall eines Gesichts) an der Bildposition (i_0, j_0) findet dann statt durch

$$(i_0, j_0) = \underset{i,j}{\operatorname{argmax}} \left\{p(x_{(i,j)}|k)\right\} \tag{3.3}$$

Die beiden Wahrscheinlichkeitsverteilungen $p_F(x|k)$ und $p_{\bar{F}}(x|k)$ werden in [21] mittels GAUSS'scher Mischverteilungen modelliert. Um zusätzlich verschiedene Skalierungsstufen abdecken zu können, wird ein Multiskalenansatz gewählt, indem die sliding-window-Prozedur für verschiedene Vergrößerungsstufen durchgeführt wird. Bei Experimenten in [21] hat sich die Lokalisierung von Gesichtern mittels (3.1) gegenüber template matching unter Verwendung der euklidischen Distanz und einem nur auf dem Maß DFFS basierenden Verfahren als leistungsfähiger erwiesen, sowohl was die Klassifikationsrate als auch was die false-alarm-Rate betrifft.

3.2 Zu erwartende Probleme

In diesem Abschnitt sollen kurz die durch die (im Vergleich zu Kapitel 2) veränderte Aufgabenstellung zu erwartenden Probleme motiviert werden.

Lokalisierung

Während bei der USPS-Datensammlung (siehe Abschnitt 2.2) die Objekte (d.h. Ziffern) weitgehend formatfüllend in einem Bild vorhanden sind, müssen Objekte in allgemeinem Bildmaterial zunächst lokalisiert werden. In der vorliegenden Arbeit soll die Lokalisierung direkter Bestandteil der Klassifikation sein, d.h. es soll auf einen vorgeschalteten Segmentierungsschritt verzichtet werden. Dieses Vorgehen (Integration von Segmentierung und Klassifikation) hat sich auch im Bereich der Spracherkennung bewährt. Ferner ist zu beachten, daß eine fehlerhafte Lokalisierung von Objekten auch direkten Einfluß auf das Klassifikationsresultat haben kann. Dies wird später diskutiert werden, an dieser Stelle sei lediglich auf Abbildung 3.10, Seite 66 verwiesen.

Objekttransformationen

Die Datensammlung USPS enthält bezüglich Translation und Skalierung weitgehend normierte Bilder in dem Sinne, daß die Ziffern stets formatfüllend in einem 16x16 Pixel großen Bild liegen. Darüber hinaus sind die Ziffern gegenüber Rotation ebenfalls grob normiert, was sich im Rahmen durchgeführter Experimente und auch visuell bestätigt, betrachtet man z.B. die gleiche

Ausrichtung aller Einsen. Bei der Suche nach Objekten in Bildern muß überlegt werden, welchen Transformationen die zu suchenden Objekte unterworfen sein können. Dementsprechend ist die Suche zu organisieren. Verschiedene Arbeiten beschäftigen sich nicht nur mit der Lokalisierung oder Klassifikation von Objekten in Bildern, sondern zielen auch darauf ab, festzustellen, welchen Transformationen das im Bild gefundene Objekt unterliegt, insbesondere, falls es sich um ein dreidimensionales Objekt handelt. Dies wird *Lagebestimmung* (*pose estimation*) genannt. Bei manchen Ansätzen ergibt sich diese Information auch als Nebenprodukt der Klassifikation. In der vorliegenden Arbeit wird diese Aufgabe nicht behandelt, da lediglich die Klasse von im Bild enthaltenen Objekten für die Bildindizierung interessant ist.

Variable Helligkeit

Die Aufnahmen der Ziffern für die USPS-Datensammlung sind stets unter gleichen Nebenbedingungen aufgenommen bzw. nachbearbeitet worden, so daß die Beleuchtung bei allen Bildern im Korpus sehr ähnlich ist. Prinzipiell handelt es sich bei den Bildern von Ziffern sogar um binäre Informationen, was allein durch die Natur der Daten bedingt ist. Allgemeines Bildmaterial kann zu erkennende Objekte in sehr unterschiedlichen Umgebungen enthalten. Wünschenswert ist es natürlich, ein Objekt stets identifizieren zu können, unabhängig von den Umständen, unter welchen es aufgenommen wurde.

Multi-Objekterkennung bzw. leere Testszene

Bei Testszenen, die dem System präsentiert werden, ist allgemein nicht bekannt, wie viele Objekte sich in der Szene befinden bzw. ob überhaupt ein Objekt vorhanden ist.

Variabler Hintergrund

Die USPS-Datensammlung zeigt die Ziffern stets vor dem gleichen Hintergrund, das interessierende Objekt ist also stets klar als Objekt zu identifizieren. Insbesondere sind Trainings- und Testdaten stets vor dem gleichen Hintergrund aufgenommen, so daß beispielsweise bei Template Matching (siehe Abschnitt 2.4.3, Seite 22) nur Distanzen an relevanten Unterschieden zwischen Testobjekt und Referenz auftreten.

Verdeckung von Objekten

Die USPS-Datensammlung als Korpus von Ziffern zur Einzelobjektklassifikation enthält in jedem Bild eine vollständig sichtbare Ziffer. Reale Szenen mit zu lokalisierenden (und zu klassifizierenden) Objekten enthalten diese möglicherweise nur teilweise sichtbar, etwa weil sie am Bildrand liegen oder von anderen Objekten verdeckt werden.

Besonders die letzten beiden Punkte sind äußerst schwierig handhabbar, da sie ein Unterteilen einer Testszene in ‚interessante‘ und ‚uninteressante‘ Teile voraussetzen.

3.3 Die Datensammlung COIL-20 und Stand der Technik

Bei Columbia Object Image Library COIL-20[1] handelt es sich um einen Korpus aus Grauwertbildern (mit 256 Intensitätswerten) von 20 Objekten. Diese Datensammlung wurde an der Columbia University erstellt, um Experimente zu dem von Murase und Nayar in [22] dargestellten Verfahren durchzuführen. Die Erstellung dieser Datensammlung ist ferner in [24] dokumentiert.

Die Aufnahmen des Korpus zeigen Objekte, die vor schwarzem Hintergrund auf einem robotergesteuerten Drehteller aufgenommen wurden. Die Daten sind unterteilt in einen Trainingskorpus (*processed images*), der vorsegmentierte Bilder der Größe 128x128 zu den 20 Objekten umfaßt, und einen Testkorpus (*unprocessed images*), der nichtsegmentierte Aufnahmen der Größe 448x416 von 5 der 20 Objekte des Trainingskorpus umfaßt[2]. Trainings- und Testkorpus weisen unterschiedliche Beleuchtungen auf.

Die Aufnahmen eines Objekts bestehen jeweils aus einer 360° Rotations-Bildsequenz mit Hilfe des Drehtellers. Eine Rotationssequenz ist in 72 Schritte (also 5°) unterteilt, d.h. für jede Objektklasse sind 72 Bilder vorhanden, was insgesamt 1440 Trainingsbilder ergibt. Abbildung 3.2 zeigt die 20 Objekte der Trainingsdaten. Zwecks Vergleichbarkeit ist bei allen Objekten das Bild zum gleichen Drehwinkel dargestellt.
Die Trainingsdaten wurden stets so segmentiert, daß sie formatfüllend in dem 128x128 Pixel großen Bildausschnitt liegen. Abbildung 3.3 zeigt Objekt 1 in allen 72 Positionen.
Beispiele zu den unsegmentierten Testbildern sind in Abbildung 3.4 dargestellt. Zu beachten ist, daß die Klassifikationsaufgabe in der vorliegenden Arbeit weiterhin als 20-Klassen-Problem behandelt wird, d.h. die Tatsache, daß 15 Klassen in den Testdaten nicht auftreten, wird nicht vom Erkennungssytem benutzt.

Zwar verfolgt diese Diplomarbeit nicht exakt die gleichen Ziele wie Murase und Nayar in [22], allerdings werden Teile dieser Datensammlung auch in anderen Arbeiten verwendet, beispielsweise in [30], [27] und [1]. In dortigen Experimenten wird allerdings mit einer Aufteilung des *processed*-Korpus von COIL-20 in Trainings- und Testdaten gearbeitet.

Im folgenden sollen die Untersuchungen von verschiedenen Forschergruppen, welche COIL-20 (oder Teile davon) für Experimente benutzt haben, kurz vorgestellt werden. Eine unmittelbare Vergleichbarkeit von Ansätzen oder Ergebnissen mit dieser Diplomarbeit ist - anders als in Kapitel 2 - nur eingeschränkt möglich:

- Murase und Nayar konstruieren in [22] ein Echtzeit-Erkennungssystem für COIL-20, welches mit Segmentierung arbeitet. Ferner ist es ihr Ziel, den Drehwinkel für das Objekt im Testbild

[1] `http://www.cs.columbia.edu/CAVE/research/softlib/coil-20.html`

[2] In [24] werden 10 Objekte, also 720 Bilder, genannt.
Unter `http://www.cs.columbia.edu/CAVE/research/softlib/coil-20.html` sind jedoch lediglich 360 Bilder von 5 der 20 Objekte vorhanden.

Abbildung 3.2: Die 20 Objekte der Datensammlung COIL-20 (Referenzdaten)

zu ermitteln. Der von den Autoren verwendete Testkorpus ist nicht verfügbar. Es wird eine Fehlerrate von 0% erzielt.

- Baker und Nayar stellen in [1] ein Konzept vor, das hauptsächlich der Beschleunigung von Klassifikationsvorgängen bzw. von Suchen nach interessanten Bildregionen dient. Die Experimente der Autoren finden auf dem in ‚Trainings'- und ‚Testdaten' unterteilten COIL-Trainingskorpus (*processed*) statt, d.h. es findet lediglich eine normale Klassifikation von Einzelobjekten (siehe Kapitel 2) statt. Die Autoren erzielen eine Fehlerrate von 0%.

- Pösl und Niemann betrachten in [27] lediglich ein Lokalisierungsproblem (verbunden mit der Bestimmung des Drehwinkels für das zu lokalisierende Objekt). Die Klassifikation wird

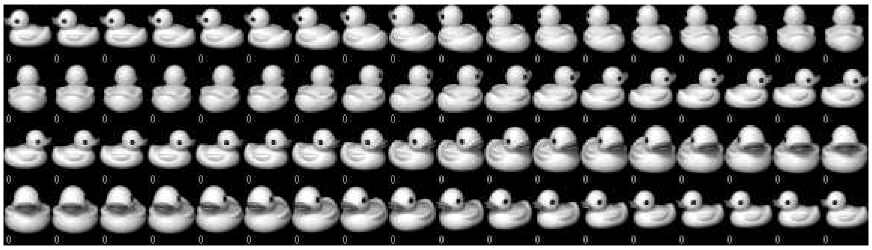

Abbildung 3.3: Alle Trainingsaufnahmen von Objekt 1 der Datensammlung COIL-20

nicht durchgeführt. In Experimenten wird lediglich ein Objekt der COIL-20-Datensammlung verwendet, wobei auch hier Trainings- und Testdaten aus dem COIL-20 Trainingskorpus generiert werden. Für Testzenen werden die Objekte zum Teil mit anderer Beleuchtung und inhomogenem Hintergrund versehen.

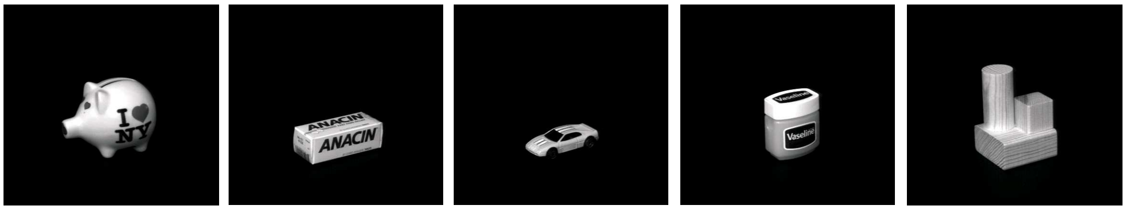

Abbildung 3.4: Beispiele für Testbilder in COIL-20

Das Ziel von Murase und Nayar in [22] ist es, ein Echtzeitsystem zu erstellen, welches in der Lage ist, *erscheinungsbasiert* die Klasse (classification) und Lage (pose estimation) von Objekten vor homogenem Hintergrund festzustellen. Dabei sollen die Objekte ferner vollständig sichtbar sein.

Der erscheinungsbasierte Ansatz bedeutet wie in Kapitel 2, daß die Daten, insbesondere diejenigen, welche zum Training des Systems benutzt werden, aus den Grauwertbildern an sich bestehen und keine anderen Repräsentation - wie bei Bildern von dreidimensionalen Objekten beispielsweise CAD-Modelle - verwendet werden. Die Nebenbedingung, daß die Objekte stets vor schwarzem Hintergrund aufgenommen sind, macht die Lokalisierung eines 3D-Objekts in einem Bild sehr leicht. Da ein Objekt ferner stets vollständig sichtbar ist, wird in [22] der Übergang vom Grauwertbild eines Objekts zu dessen *Eigenraum-Repräsentation* vorgeschlagen, d.h. Anwendung der Karhunen-Loève-Transformation zur Merkmalsreduktion. Dies kann (neben der allgemeinen Eignung der KLT zur - verlustbehafteten - Datenkompression) dadurch motiviert werden, daß zwei benachbarte Bilder einer Bildsequenz sehr ähnlich zueinander sind, d.h. eine hohe Korrelation besitzen. Hierbei ist allerdings zu beachten, daß dies nur für benachbarte Bilder gilt, während die Variabilität der Grauwerte innerhalb der gesamten Bildsequenz unter Umständen sehr hoch ist. Deshalb ist es nach Nayar und Murase auch nicht sinnvoll, ein Clustering der dimensionsreduzierten Daten zu betrachten. Dies ist in [22] ferner auch nicht erwünscht, da das zweite Ziel (neben der Klassifikation) die Bestimmung der Lageparameter für ein erkanntes 3D-Objekt ist. Stattdessen wird die merkmalsreduzierte Bildsequenz im Eigenraumen nach den sich innerhalb der Sequenz ändernden Lageparametern parametrisiert, was über Interpolation mittels kubischer Splines geschieht. Dieses Konzept wird als *parametric eigenspace* (parametrisierter Eigenraum) bezeichnet. Interessant ist eine sinnvolle Interpolation auch im Hinblick auf die Klassifikation von Bildern mit in den Trainingsdaten nicht gesehenen Lageparametern für ein Objekt. Die Interpolation ist relativ gut möglich, da aufgrund der schon erwähnten hohen Korrelation benachbarter Sequenz-Bilder auch deren Abstände im Eigenraum gering sind. Die Lageparameter bei den Experimenten in [22] sind Drehwinkel des Tellers und ein Parameter für die Beleuchtung[3] der Szene. Die

[3]Der verfügbare COIL-20-Korpus besitzt keine Trainingssequenzen für diesen Parameter. Es ist lediglich eine Rotationssequenz pro Objekt vorhanden. Die Helligkeit ist hierbei konstant. Die Testszenen sind unter anderen

Klassifikation und die Lagebestimmung eines Testbildes erfolgt durch Transformation des Testbildes den Eigenraum und die Bestimmung des kürzesten Abstands des transformierten Bildes zu einer der durch die Trainingssequenzen definierten Mannigfaltigkeiten im Eigenraum. Dies löst gleichzeitig das Klassifikationsproblem (durch Bestimmung der Mannigfaltigkeit und damit der Objektklasse) und das Lagebestimmungsproblem (durch Bestimmung der Transformationsparameter des zum transformierten Bild nächstgelegenen Elementes der Mannigfaltigkeit). Die Bestimmung des nächstgelegenen Elementes der Mannigfaltigkeit ist allerdings ein schwieriges Problem, für das in [22] verschiedene Algorithmen verwendet werden, u.a. auch ein Neuronales Netz.

Um höhere Genauigkeit bei der Lagebstimmung zu erreichen, wird diese im klassenabhängigen Eigenraum ausgeführt, d.h. die Schätzung der Merkmalsreduktion hierfür findet pro Klasse getrennt statt. Die Wahl des Eigenraums findet nach der Klassifikation statt, die weiterhin im globalen Eigenraum stattfindet. Insgesamt ergibt sich für das in [22] dargestellte Verfahren der in Abbildung 3.5 skizzierte Ablauf.

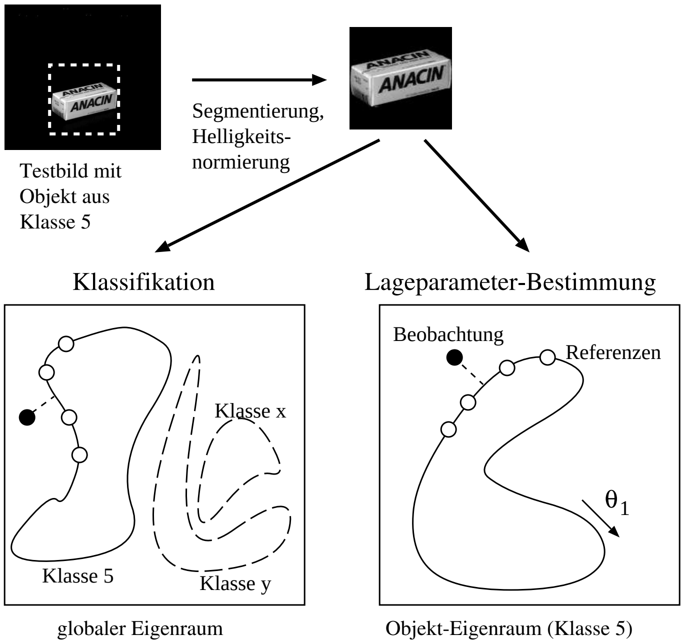

Abbildung 3.5: Funktionsweise eines Echtzeit-Erkennungssystems für COIL-20 nach [22]. Zu beachten ist der Verzicht auf Clustering der Daten im Eigenraum. Stattdessen werden die Trainingsvektoren einer Sequenz nach den Lageparametern parametrisiert und die Mannigfaltigkeit mittels Interpolation approximiert. Die interpolierten Mannigfaltigkeiten sind im dargestellten Fall geschlossene Kurven, was z.B. für eine 360° Rotationssequenz der Fall ist.

Ein Testbild wird segmentiert und das lokalisierte Objekt auf 128x128 Pixel skaliert. Anschließend wird eine Helligkeitsnormierung durchgeführt, indem die Energie des Bildes auf 1 normiert wird. Das Bild wird darauf in einen 20-dimensionalen Eigenraum (konstruiert aus allen 1440 Trainingsbil-

Beleuchtungen aufgenommen.

dern der 20 Klassen) projiziert. Bei Experimenten in [22] wird allerdings auf klassenspezifische Eigenräume zur Lagebestimmung verzichtet und diese ebenfalls im globalem Eigenraum durchgeführt. Der Eigenraum wird nach dem Drehwinkel parametrisiert. Getestet wird das System mit 320 Aufnahmen, für die zufällig Objekte und Drehwinkel ausgewählt wurden (dies sind offenbar nicht die Aufnahmen, welche unter `http://www.cs.columbia.edu/CAVE/research/softlib/coil-20.html` verfügbar sind. Bei den verfügbaren handelt es sich um komplette, unsegmentierte Rotationssequenzen für 5 der 20 Objekte, welche ferner unter anderen Beleuchtungsverhältnissen als die Trainingsdaten aufgenommen wurden. Insgesamt sind dies 360 Bilder). Das System in [22] führt Segmentierung, Normierung, Eigenraum-Projektion, Klassifikation und Lagebestimmung innerhalb einer Sekunde aus und kann alle 320 Testbilder korrekt klassifizieren. Die mittlere Abweichung der Drehwinkelbestimmung liegt bei 1.5°.

Die Arbeiten [27] und [28] von Pösl und Niemann stellen ein Verfahren vor, mit dem ein Objekt in variabler Translation, Rotation und Beleuchtung auch in Szenen mit inhomogenem Hintergrund und teilweiser Verdeckung des Objekts lokalisiert werden kann. Dort werden hierarchische Gabor-Filter benutzt, um lokale Merkmale aus dem Bild zu extrahieren. Diese sind (nach Konstruktion) invariant gegenüber Rotation und robust gegenüber variabler Beleuchtung und Rauschen. Eine Multiskalen-Hierarchie für die Filterung wird verwendet, um den Suchraum bei der Lokalisierung eines Objekts in einer Testszene einzuschränken: Mit zunehmender Auflösung werden so nur noch vielversprechende Bildregionen abgesucht. Die Suche erfolgt durch Überlagerung des zu durchsuchenden Bildes mit einem Gitter diskreter Positionen, an denen jeweils die lokalen Merkmale berechnet werden. Die Merkmale einer Region A auf dem Gitter werden zu einem Merkmalsvektor x_A konkateniert (der Index s für die aktuell betrachtete Hierachiestufe wird weggelassen) und für diesen eine Wahrscheinlichkeit $p(c_A|B,R,t)$ berechnet. Hierbei repräsentiert B die Parameter des statistischen Modells für p, während R (Rotation) und t (Translation) die Transformationen des Bildausschnitts c_A darstellen. Die Schätzung der Lage eines Objekts in einer Szene erfolgt dann durch (man beachte, daß keine Klassifikation durchgeführt wird, d.h. die Klasse des zu lokalisierenden Objekts ist bekannt):

$$(\hat{R},\hat{t}) = \underset{(R,t)}{\operatorname{argmax}} \{p(c_A|B,R,t)\} \tag{3.4}$$

Bei den Transformationen wird unterschieden zwischen *internen*, d.h. solchen, welche die Bildebene nicht verlassen und *externen*, welche dies tun. Abbildung 3.6 veranschaulicht interne und externe Transformationen.

Die erste Menge wird bei der Szenenanalyse explizit abgesucht, was bei Rotation ϕ_z und Translation t_{2D} in der Bildebene einen dreidimensionalen Suchraum ergibt. Dies ist in Abbildung 3.7 dargestellt.

Die zweite Menge, also Transformationen, welche die Bildebene verlassen, wird statistisch modelliert, d.h. für diese müssen Trainingsdaten vorhanden sein. Beim COIL-20 Korpus ist dies die 360° Rotationssequenz eines Objekts.

Die Modellierung von $p(c_A|B,R,t)$ wird in [27] über eine Gauss'sche Normalverteilung durchgeführt. Für das statistische Modell ergibt sich somit (man beachte, daß die Skalierung eines

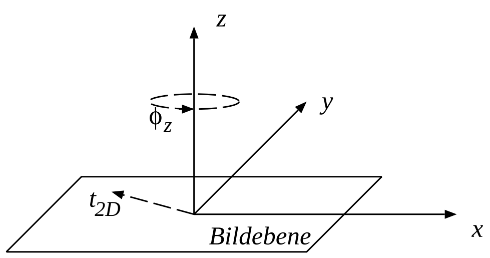

Abbildung 3.6: Veranschaulichung interner und externer Transformationen eines Bildes nach [27]. Die Transformationen t_{2D} und ϕ_z sind interne Transformationen, da durch sie die Bildebene nicht verlassen wird.

Objekts als fest angenommen wird, d.h. $t_z = 0$):

$$\begin{aligned} p(c_A|B,R,t) &= p(c_A|(\mu(\phi_y,\phi_x),\Sigma(\phi_y,\phi_x)),R,t) && (3.5)\\ &= \mathcal{N}(c_A(\phi_z,t_{2D})|\mu(\phi_y,\phi_x),\Sigma(\phi_y,\phi_x)) && (3.6)\end{aligned}$$

Zu beachten sind ferner die Nebenbedingung des Systems, daß die Klasse des Objekts bei der Szenenanalyse bekannt ist. Diese Einschränkungen lassen sich zwar konzeptionell recht leicht auflösen, jedoch wird dies nicht ausgeführt. Es ist unklar, welche Leistung das System dann erbringt. Ferner ist dem System bekannt, daß stets genau ein Objekt in der Szene vorhanden ist.

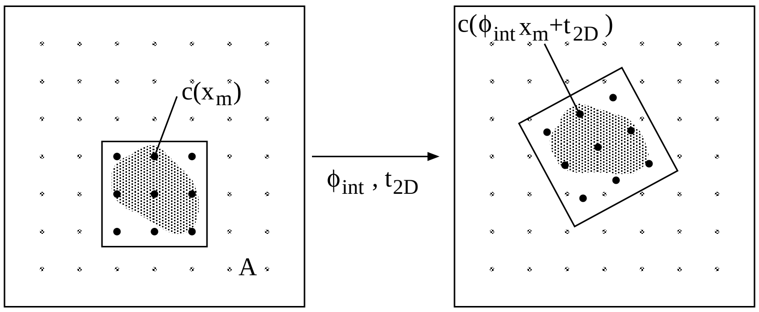

Abbildung 3.7: Suche im Parameterraum interner Transformationen nach [27]: Das Gitter wird nacheinander in verschiedenen Winkeln ϕ_z über die Testszene gelegt und mit einem Sliding Window (dargestellt durch A) durchsucht. Durch die Verwendung hierarchischer Gabor-Filter (die Hierarchie kann als Schrittweite des Gitters interpretiert werden) kann der Parameterraum effizienter durchsucht werden, indem nur erfolgversprechende Hypothesen beim Abstieg in der Hierarchie weiterverfolgt werden.

Um die Lokalisierung robust gegenüber variablem Hintergrund zu machen, wird in [27] eine explizite Hintergrundmodellierung durchgeführt, welche Wahrscheinlichkeiten für die Zugehörigkeit des aktuell betrachteten Ausschnitts zu den Klassen ‚Objekt' und ‚Hintergrund' darstellt. Für den Hintergrund entfallen hierbei die für Objekte verwendeten Lageparameter im Modell. Die Wahrscheinlichkeit für die Zuordnung eines Ausschnitts zu Vorder- bzw. Hintergrund wird in lokaler Abhängigkeit zu Nachbarpositionen modelliert. Bei Experimenten in [27] wird eine Versuchsreihe mit der Klasse *Sparschwein* (siehe Abbildung 3.2, Seite 56) aus dem COIL-20-Korpus durchgeführt, wobei die 72 Bilder des *processed*-Korpus in Trainings- und Testdaten aufgeteilt wurden (mit jeweils 36 Bildern). Die Testszenen bestehen aus inhomogenem Hintergrund, vor den jeweils ein Testobjekt eingesetzt wurde. Der genaue Aufbau der Testszenen ist etwas unklar. PÖSL und NIEMANN geben an, daß auch die Menge der internen Rotationen (d.h. Rotationen in der Bildebene) bei der Analyse einer Szene abgesucht wird, d.h. auch dieser Parameter scheint in den Testszenen zu variieren. Die Lokalisierung bei inhomogenem Hintergrund gelingt in allen Testfällen.

Obwohl sich die Arbeit [30] von SCHIELE lediglich mit der Einzelobjektklassifikation (siehe Kapitel 2) befaßt und den COIL-Trainingskorpus hierbei zur Evaluation heranzieht, wird in dieser Arbeit ein Verfahren benutzt, welches SCHIELE und CROWLEY in [31] zur Lokalisierung von Objekten in Testszenen verwenden. Es handelt sich hierbei um die Anwendung von *multdimensional receptive field histograms*. Grundprinzip ist es, ein Merkmalshistogramm einer Region der Testszene mit dem einer Referenz zu vergleichen und auf diese Art Objekte zu lokalisieren (d.h. es wird mit lokalen Merkmalen gearbeitet). Der Ansatz besteht aus

- der Wahl geeigneter lokaler Merkmale,
- einem Mechanismus zum Vergleich zweier Histogramme,
- dem geschickten Aufbau der Histogramme bzüglich Merkmalsanzahl und Quantisierung der Histogrammeinträge

Als lokale Merkmale werden *Receptive fields* verwendet. Dies sind lokale lineare Operationen, die auf der Nachbarschaft eines Pixels ausgeführt werden, um lokale Form-Merkmale zu gewinnen. Dies erfolgt in [31] durch Anwendung gerichteter Gradientenfilter. Diese werden anschliessend energienormiert. Pro betrachteten Bildausschnitt werden N ‚Messungen' $M_1, \ldots M_N$ lokaler Formmerkmale vorgenommen. Die Objekterkennung wird als Maximum-Likelihood-Entscheidung ausgeführt, wobei die statistische Unabhängigkeit der bestimmten Merkmale angenommen wird:

$$p(k|\bigwedge_n M_n) = \operatorname*{argmax}_k \left\{ \prod_n p(M_n|k) \right\} \tag{3.7}$$

Die Wahrscheinlichkeit $p(M_n|k)$ ist jeweils durch das *multdimensional receptive field histogram* der Messung M_n gegeben.

3.4 Aufbau des Bildindizieres

Grundprinzip des Bildindizieres in der vorliegenden Arbeit ist es, mit einem *sliding window* fester Größe über das präsentierte Testbild zu fahren und den sich momentan im Fenster befindenden Bildausschnitt mit Methoden wie in Kapitel 2 dargestellt zu klassifizieren. Mit diesem Ansatz kann bereits die variable Translation eines Objekts abgedeckt werden. Um ferner verschiedene Skalierungsstufen abzudecken, wird ein Multiskalenansatz gewählt, d.h. die sliding-window-Prozedur wird für verschiedene Vergrößerungsstufen des Testbildes ausgeführt. Analog zu [21] entstehen so von MOGHADDAM und PENTLAND mit *multiscale saliency maps* bezeichnete Karten des Testbildes,

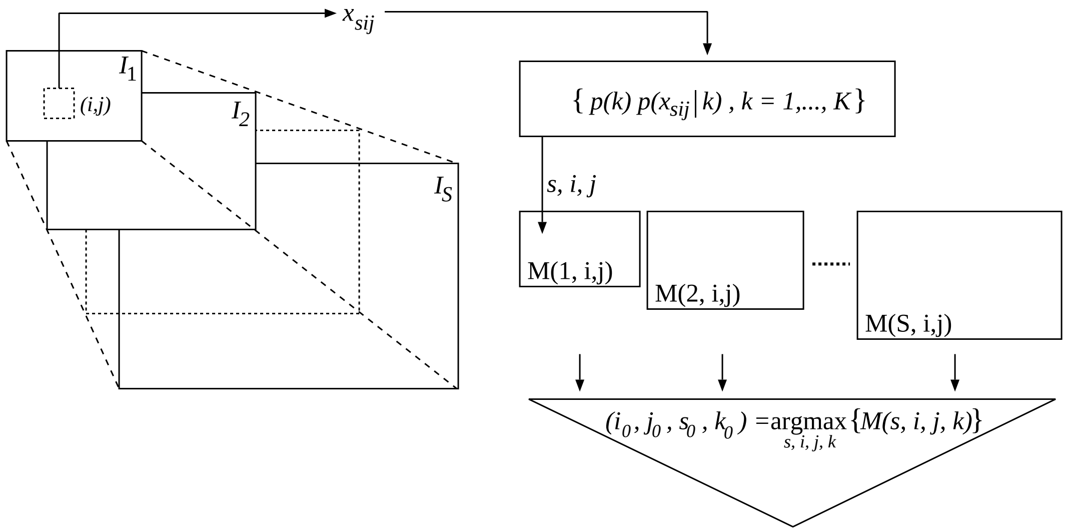

Abbildung 3.8: Multiskalenansatz mit sliding window zur Einzelobjektsuche: Mit dem Abfahren der Testszene mit einem sliding window in verschiedenen Auflösungen entstehen die Karten $M(s,i,j), s = 1,\ldots,S$. Diese enthalten zu jeder Bildposition eine Klassifikationshypothese für den Inhalt der mit der Bildposition (i,j) assoziierten Region des skalierten Bildes I_s. Vergleiche [21].

die für jede Position in einer Menge von Auflösungsstufen Wahrscheinlichkeiten für Objektklassen-Zugehörigkeit des jeweiligen Bildausschnitts beeinhalten. Diese Karte muß anschließend in geeigneter Art und Weise ausgewertet werden. Das Vorgehen des Systems ist in Abbildung 3.8 dargestellt.

Anders als bei der Situation in Kapitel 2 ist es möglich, daß der aktuell betrachtete Bildausschnitt nicht das enthält, was als Objekt bezeichnet werden sollte, also z.B. nur Teile davon (weil momentan Position oder Skalierung nicht ‚passen') oder Hintergrund. Es muß also ein Mechanismus vorhanden sein, um Entscheidungen eines zu Kapitel 2 analogen Klassifikators, der sich immer für eine Objektklasse entscheidet, abzulehnen. Dies soll mit *reject* bezeichnet werden.

3.4.1 Lokale Entscheidungen

Der Bildindizierer hat für eine Testszene variable Antwortmöglichkeiten, da die Szene eins, mehrere, oder auch kein Objekt enthalten kann. Deshalb ist es schwierig, die für den Bildindizierer verwendete Entscheidungsfunktion $r(I)$ kompakt darzustellen. Geht man davon aus, daß genau ein Objekt in Testbild I vorhanden ist, so erhält man

$$r(I) = \operatorname*{argmax}_k \left\{ \max_{s,i,j} \{p(k|x_{sij})\} \right\} \tag{3.8}$$

wobei x_{sij} der in Skalierungsstufe s aus I_s an Position (i,j) ausgeschnittene Merkmalsvektor ist:

$$x_{sij} = \downarrow [\{I_s(i+r, j+c); (r,c) \in R\}] \tag{3.9}$$

Das sliding window sei durch die Indexmenge R, $|R| = D$ repräsentiert und $\downarrow [.]$ die Funktion, welche die ausgeschnittenen Grauwerte in einen Merkmalsvektor $x \in I\!R^D$ transformiert.
Die Wahrscheinlichkeit $p(k|x_{sij})$ wird zunächst unabhängig von der aktuell betrachteten Position (i,j) oder Skalierungsstufe s modelliert, d.h. falls $x_{sij} = x_{s'i'j'}$ gilt, so ist $p(k|x_{sij}) = p(k|x_{s'i'j'})$. Bei späterer Verwendung zusätzlicher Informationen ausserhalb des aktuellen Bildausschnitts ist dies unter Umständen nicht mehr der Fall. Bei der Modellierung der Wahrscheinlichkeit $p(k|x_{sij})$ stehen prinzipiell die in Kapitel 2 dargestellten Ansätze zur Verfügung.

3.4.2 Konfidenz in eine lokale Entscheidung

In Formulierung (3.8) wird nicht berücksichtigt, wie vertrauenswürdig die Entscheidung für eine Objektklasse ist. An sehr vielen Bildpositionen wird die Entscheidung für eine Klasse unsinnig sein. Dementsprechend sollte für jeden Merkmalsvektor x_{sij} eine Wahrscheinlichkeit vergeben werden, daß x_{sij} tatsächlich ein Objekt darstellt. Diese relativiert die lokale Entscheidung des Klassifikators. Liegt die Konfidenz unterhalb eines Schwellenwertes, wird der aktuelle Bildausschnitt als Position eines Objektes verworfen (reject). Somit wird (3.8) modifiziert zu

$$\begin{aligned} r(I) &= \operatorname*{argmax}_k \left\{ \max_{s,i,j} \{p(k, x_{sij} \text{ ist Objekt}|x_{sij})\} \right\} \\ &= \operatorname*{argmax}_k \left\{ \max_{s,i,j} \{p(k|x_{sij} \text{ ist Objekt}, x_{sij}) \cdot p(x_{sij} \text{ ist Objekt}|x_{sij})\} \right\} \\ &\overset{Modell}{=} \operatorname*{argmax}_k \left\{ \max_{s,i,j} \{p(k|x_{sij}) \cdot p(x_{sij} \text{ ist Objekt}|x_{sij})\} \right\} \end{aligned} \tag{3.10}$$

Hierbei wird in (3.10) angenommen, daß die Ereignisse (x_{sij} ist Objekt) und (x_{sij} ist Objekt der Klasse k) voneinander unabhängig sind. Die Entscheidung $p(x_{ijk}$ ist Objekt$|x_{sij})$ kann distanzbasiert fallen, beispielsweise -im Falle einer Nearest-Neighbour-Klassifikation- als Normalverteilung der auftretenden Distanzen

$$p(x_{sij} \text{ ist Objekt}) \overset{Modell}{=} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2}\frac{(d_{\text{norm}}(x_{sij}) - \mu)^2}{\sigma^2}\right], \text{ wobei} \tag{3.11}$$

$$d_{\text{norm}}(x_{sij}) = \frac{\min\limits_{n,k} \{d(x_{sij}, x_{nk})\}}{\max\limits_{s',i',j'} \left\{ \min\limits_{n,k} \{d(x_{s'i'j'}, x_{nk})\} \right\}} \tag{3.12}$$

Hierbei wird der Mittelwert $\mu = 0$, d.h. die theoretisch minimale Distanz einer Beobachtung zu einer Referenz, verwendet. Die Menge der Trainingsdaten ist dargestellt durch $\{x_{nk}, n = 1, \ldots, N_k, k = 1, \ldots, K\}$. Die Varianz σ^2 ist je nach Verteilung der Distanzen einzustellen. In den Experimenten wird $\sigma^2 = \frac{1}{2\pi}$ benutzt, damit $p(x_{sij}$ ist Objekt$) = 1$ gilt, falls $d_{\text{norm}}(x_{sij}) = 0$. (3.12) skaliert die auftretenden Distanzen linear auf einen Wertebereich zwischen 0 und 1. Der Verlauf der Konfidenz abhängig von der erzielten Distanz (3.11) ist in Abbildung 3.9 dargestellt. Verwendet man zusätzlich einen Schwellenwert t in (3.10), so kann auch ausgedrückt werden, daß das Bild nach Meinung des Klassifikators kein einziges Objekt enthält. Die Entscheidung (3.10) wird so modifiziert zu

$$r(I) = \begin{cases} r(I) \text{ gemäß (3.10);} & \text{falls } p(x_{sij} \text{ ist Objekt}) > t \\ reject & \text{sonst} \end{cases} \quad (3.13)$$

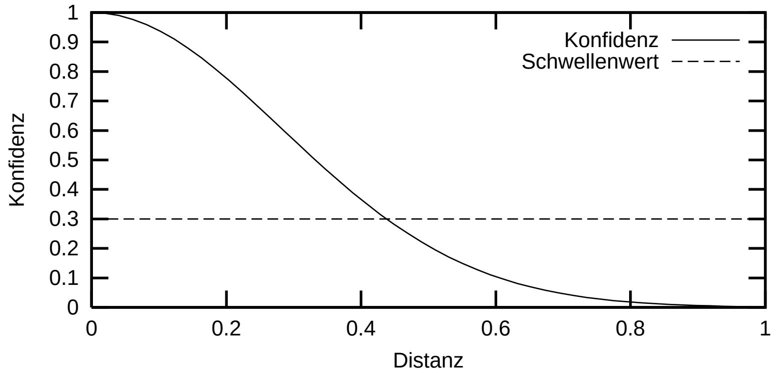

Abbildung 3.9: Konfidenz in eine lokale Entscheidung in Abhängigkeit der erzielten Distanz. Der Wertebereich der für die Bildausschnitte der Testszene auftretenden Distanzen wird linear auf $[0 \ldots 1]$ skaliert. Mit einem Schwellenwert (hier 0.3) können Bildausschnitte, bei denen die Konfidenz zu gering ist, als Objektposition verworfen werden.

3.4.3 Multi-Objekterkennung

Die Erkennung mehrerer Objekte erfolgt durch iterative Anwendung von Kriterium (3.13), wobei für jedes erkannte Objekt der von ihm belegte Bildbereich aus der Karte von I in den folgenden Schritten nicht mehr berücksichtigt wird, d.h. in Pseudocode:

```
(1)  while TRUE do
(2)          wende (3.13) an
(3)          if ( kein weiterer Treffer )
(4)                  exit
(5)          entferne den Treffer aus der Karte
(6)  done
```

Schritt (3) prüft, ob (3.13) mit ‚reject' geantwortet hat. Das Entfernen eines Treffers aus der Karte in Schritt (5) findet in der vorliegenden Arbeit statt, indem der von einem Treffer belegte Bildbereich in der Karte in darauffolgenden Suchen in der Karte ignoriert wird. Dies geschieht aus Gründen der Rechenzeit. Genauer ist es, nach der Bestimmung eines Treffers die Karte unter der Voraussetzung bisher erzielter Treffer neu aufzubauen.

3.4.4 Motivation des weiteren Vorgehens

In der vorliegenden Arbeit wird für die ausgeschnittenen Merkmalsvektoren der Nearest Neighbour Klassifikator verwendet. Dieser wurde aus folgenden Gründen gewählt:

- Visuelle Kontrolle des Klassifikationsergebnisses durch Ansicht der passenden Referenz.

- Durch Verwendung der Tangentendistanz wird der Nearest Neighbour zu einem konkurrenzfähigen Klassifikator. Durch den Einsatz der Skalierungstangente kann gehofft werden, die Pyramide beim Multiskalenansatz besser aufzulösen.
- Anwendung intuitiv verständlicher Methoden der Bildverarbeitung. Da beim Nearest Neighbour auf Pixel- und Grauwertebene gearbeitet wird, können in die zu berechnenden Distanzen weitere Informationen integriert werden, die sich z.B. auf Bereiche außerhalb des aktuellen Bildausschnitts beziehen. Bei anderen Methoden (z.B. Merkmalsreduktion) ist dies nicht ohne weiteres möglich.
- Eine statistische Modellierung von COIL-20 mit GAUSS'schen Mischverteilungen ist aufgrund der geringen Menge an Trainingsdaten, aber auch aufgrund der Art des Datenmaterials, ungünstig, da nur eine Ansicht pro Drehwinkel (was einer Dichte entsprechen könnte) vorhanden ist.

Durch den Multiskalenansatz in Verbindung mit einem sliding window ergeben sich erhebliche Rechenzeiten für das Absuchen eines Bildes. Hierbei ist zu überlegen, ob und wie bestimmte Regionen, in denen sich keine Objekte befinden, als solche erkannt werden können. BAKER und NAYAR nennen diese Voruntersuchung *Pattern Rejection* [1]. Dort wird eine Funktion, welche für einen Merkmalsvektor die Anzahl in Frage kommender Klassenzuordnungen verringert oder den Merkmalsvektor verwirft, als *Rejector* bezeichnet. Für die COIL-20-Datensammlung, deren Testszenen stets Einzelobjekte vor schwarzem Hintergrund enthalten, bieten sich folgende Möglichkeiten an:

1. Verwerfen von Bildausschnitten x_{sij}, deren Grauwertmasse $m(x_{sij}) = ||x_{sij}||^2$ einen Schwellenwert unterschreitet. Diese Ausschnitte sind als Hintergrund einzustufen.
2. Verwerfen von Bildausschnitten, bei denen sich ausserhalb des Ausschnitts deutlich mehr Grauwertmasse als innerhalb befindet. In diesen Fällen ist davon auszugehen, daß der aktuell betrachtete Ausschnitt nicht das gesamte Objekt umfaßt, sondern lediglich einen (zu kleinen) Teil des Testobjekts ‚erklärt'. Dies ist in Abbildung 3.10 dargestellt. Die Grauwertmasse außerhalb des aktuell betrachteten Bildausschnitts wird im folgenden *Handicap* genannt. Der in dieser Arbeit implementierte Rejector für das Handicap wertet das Verhältnis zwischen Grauwertmasse innerhalb und Grauwertmasse außerhalb des aktuell betrachteten Bildausschnitts aus. Wird im folgenden von einer Handicap-Reject Einstellung von 1:0.1 gesprochen, ist damit gemeint, daß alle Bildausschnitte durch den Rejector verworfen werden, bei denen die Grauwertmasse außerhalb des momentan betrachteten Bildbereichs mehr als 10% der Grauwertmasse des aktuellen Bildausschnitts beträgt.
3. Verwerfen von Bildausschnitten, die -anhand grober Kriterien abgelesen- nicht zu den Referenzen ‚passen'. Kriterien können hierbei die Grauwertmasse m oder die Entropie H [18, Abschnitt 4.3.2] von x_{sij} sein. Schwellenwerte t_{min}, t_{max} zu diesen Größen, z.B. für die Entropie, können aus den Trainingsdaten ermittelt werden:

$$t_{min} = (1-\alpha) \cdot \min_{n,k} \{H(x_{nk})\} \qquad (3.14)$$

$$t_{max} = (1+\alpha) \cdot \max_{n,k} \{H(x_{nk})\} \tag{3.15}$$

Hierbei kann ein Toleranzfaktor $0 < \alpha < 1$ gewählt werden. Wird dieser Punkt angewendet, deckt er die Funktionalität von Punkt 1 mit ab.

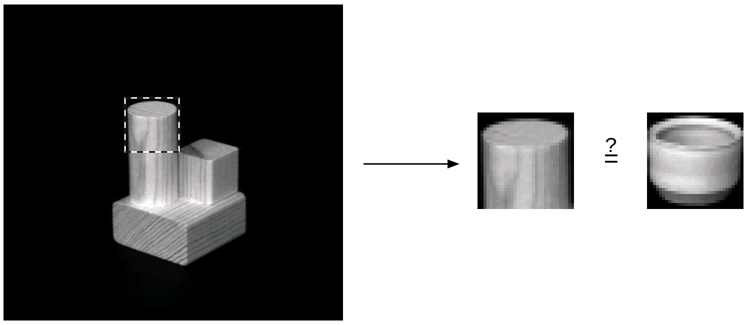

Abbildung 3.10: Auswirkung fehlerhafter Lokalisierung auf die Klassifikation. Durch isoliertes Betrachten eines kleinen Ausschnitts werden möglicherweise Ergebnisse geliefert, die zwar lokal eine gute Erklärung der Testszene liefern, global betrachtet aber eine mangelhafte.

Motivation zum Einsatz der Entropie als Reject-Kriterium ist, daß hier starke Unterschiede zwischen Referenz und Beobachtung auf eine falsche Skalierungsstufe hindeuten können, insbesondere, was zu starke Vergrösserung betrifft. In diesem Fall treten möglicherweise relativ große monotone Flächen auf, die anhand ihrer geringen Entropie erkennbar sind.
Werden die oben genannten Punkte sehr restriktiv angewendet, ergibt sich implizit eine Segmentierung des Testbildes. Der zweite Punkt kann modifiziert werden, indem die außerhalb des aktuellen Ausschnitts liegende Grauwertmasse in die Distanzberechnung einbezogen wird, d.h. man betrachtet die Referenz mit fortgesetztem Hintergrund:

$$h_{sij} = \sum_{i',j':(i',j') \notin R(i,j)} (I_s(i',j'))^2 \tag{3.16}$$

Die Größe h_{sij}, das *Handicap* von x_{sij}, ist die Summe der Quadrate aller restlichen, nicht vom aktuellen Bildausschnitt $R(i,j)$ betrachteten Grauwerte im Bild I_s, d.h. die Distanz zu einem fiktiven Hintergrund Null (d.h. schwarz) der Referenz. Für die so modifizierte Nearest-Neighbour-Distanz zwischen einer Beobachtung x_{sij} und einer Referenz μ ergibt sich dann

$$d_{\text{handicap}}(x_{sij}, \mu) = d_{\text{NN}}(x_{sij}, \mu) + h_{sij} \tag{3.17}$$

wobei d_{NN} die normale Nearest-Neighbour-Distanz ist (siehe Abschnitt 2.4.3). Zu beachten ist, daß nun stets Distanzen über alle Pixel von I_s berechnet werden, also Distanzen in unterschiedlichen Skalierungsstufen nicht mehr vergleichbar sind. In der vorliegenden Arbeit wird daher eine Normierung der Distanzen auf die Anzahl eingegangener Pixel, d.h. die Größe von I_s, durchgeführt.

Durch Verwendung des Handicaps in der Distanzberechnung (3.17) kann erhofft werden, Probleme wie das in Abbildung 3.11 dargestellte zu vermeiden. In diesem Fall ist die für die Fehlklassifikation verantwortliche Grauwertmasse außerhalb des aktuellen Bildausschnitts gering genug, um den Ausschnitt durch einen auf Punkt 2 basierenden Rejector (für die meisten Schwellenwerte) zu passieren. Durch Einfluss dieser Grauwertmasse in die Distanzberechnung (3.17) kann ggf. eine derartige Fehlerquelle vermieden werden.

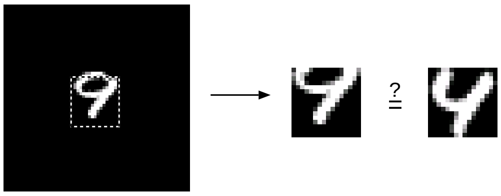

Abbildung 3.11: Auswirkung fehlerhafter Lokalisierung auf die Klassifikation. Schon quantitativ geringe Lokalisierungsfehler können das Klassifikationsergebnis beeinflussen.

Bei Verwendung des Handicap ist zu beachten, daß sich das Konzept in dieser Form nicht für die Multiobjekterkennung eignet, da die Grauwertmasse aller anderen Objekte in das Handicap eines Objekts eingeht. In der Multi-Objekterkennung wird mit einem lokalen Handicap gearbeitet, welches lediglich die Grauwertmasse in einer begrenzten Umgebung des aktuell betrachteten Bildausschnitts umfaßt. Auch dies bleibt aber problematisch, falls Objekte in einer Testszene sehr dicht zusammenliegen.

3.4.5 Helligkeitsanpassung

Ein bei der Klassifikation handgeschriebener Ziffern nicht auftretendes Problem ist die variable Helligkeit des Bildmaterials. Dies macht sich bei ersten Experimenten mit den Testbildern der COIL-Datensammlung bemerkbar.

Murase uns Nayar benutzen in [22] zur Helligkeitsnormierung die Energienormierung. Dies hat in der vorliegenden Arbeit gegenüber der lokalen Helligkeitsanpassung pro Paar (Referenz $\mu \in I\!\!R^D$, Beobachtung $x \in I\!\!R^D$) als günstiger erwiesen. Die lokale Helligkeitsanpassung kann für ein additives Beleuchtungsmodell erfolgen durch

$$\mu' = \mu + \frac{\sum_{d=1}^{D} x_d}{\sum_{d=1}^{D} \mu_d} \cdot 1^D, \text{ wobei } 1^D = (1, \ldots, 1) \in I\!\!R^D \tag{3.18}$$

bzw. für ein multiplikatives Beleuchtungsmodell durch

$$\mu' = \frac{\sum_{d=1}^{D} x_d}{\sum_{d=1}^{D} \mu_d} \cdot \mu \tag{3.19}$$

d.h. die Grauwerte einer Referenz μ werden jeweils so modifiziert, daß μ' und x über den gleichen mittleren Grauwert verfügen (der Normierungfaktor $1/D$ kürzt sich jeweils weg). Hierbei sollte zusätzlich mit Hintergrunderkennung gearbeitet werden, um die Anpassung uninteressanter Teile des Ausschnitts zu verhindern. Diese werden dann bei der Anpassung ausgeklammert. Auf COIL-20 (Bildmaterial mit schwarzem Hintergrund) kann ein Schwellenwert zur Hintergrunderkennung benutzt werden.

Allerdings werden mit der Energienormierung bessere Resultate erzielt, was sowohl die Fehlerrate als auch den Rechenbedarf betrifft. Die Energienormierung für einen Merkmalsvektor $x \in \mathbb{R}^D$ erfolgt durch

$$x \mapsto \frac{1}{||x||} \cdot x \tag{3.20}$$

In der vorliegenden Arbeit findet hierbei keine Hintergrunderkennung statt. Der Rechenaufwand ist deutlich geringer, da die Energienormierung für jeden Bildausschnitt und jede Referenz nur einmal separat ausgeführt werden muß, während die individuelle Anpassung für jedes Paar (μ, x) erfolgt.

3.4.6 Berücksichtigung von variablem Hintergrund

Betrachtet man Szenen, die Objekte vor inhomogenem Hintergrund enthalten, ist eine Identifikation von Vordergrund (d.h. interessanten Bildteilen) und Hintergrund (entsprechend uninteressante Bildteile) nicht ohne weiteres möglich, und Mechanismen wie der dargestellte Einsatz von Handicap greifen nicht mehr. Falls die Trainingsdaten immer noch homogenen Hintergrund enthalten (und dieser bestimmbar ist), können bei der Distanzberechnung Stellen, an denen die Referenz Hintergrund enthält, ignoriert werden, d.h. man läßt in den Testszenen beliebigen Hintergrund zu. Die Funktionsweise entspricht der einer Schablone, die für eine segmentierte Referenz Hintergrundpixel ausblendet. Durch dieses Verfahren gehen allerdings für jede Referenz unterschiedlich viele Pixel in die Distanzberechnung ein, was den Vergleich von Distanzen untereinander verfälscht. Daher sollte auch hier eine Normierung der Distanz auf die tatsächlich verwendete Anzahl von Pixeln bei der Distanzberechnung vorgenommen werden.

Negativ an diesem Ansatz ist, daß durch das Ausblenden des Hintergrundes in den Referenzen zu einem gewissen Grad auch die Forminformation verlorengeht. Dieser Sachverhalt ist in Abbildung 3.12 schematisch dargestellt.

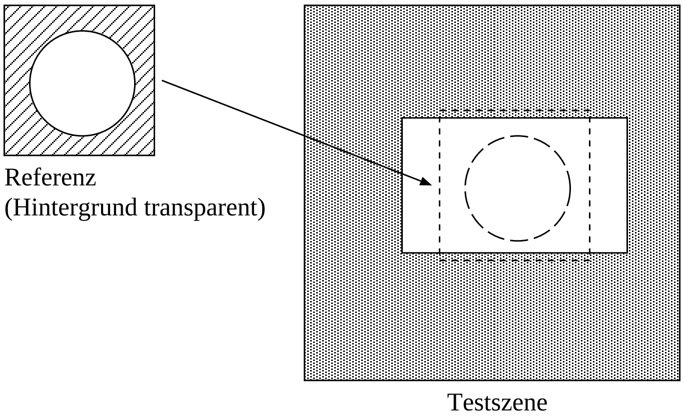

Abbildung 3.12: Problem bei Verwendung eines Transparenzkanals für die Referenzen: Die Forminformation eines Objekts geht in diesem Fall verloren. Die Distanz von Szenenausschnitt und Referenz ist Null.

Ferner ist die Normierung alleine auf die Anzahl von ‚Vordergrund'- Bildpunkten problematisch, da dies kleine Referenzen bevorzugt, welche nur einen kleineren Teil des aktuellen Bildausschnitts ‚erklären'. Bei dem mit inhomogenem Hintergrund durchgeführten Experiment in der vorliegenden Arbeit hat sich die Normierung auf die Anzahl der Vordergrund-Pixel allerdings als brauchbar erwiesen.

Moghaddam und Pentland verwenden in [21] die KLT (siehe Abschnitt 2.5.1) zur Merkmalsreduktion betrachteter Bildbereiche. Diese ist implizit tolerant gegenüber beliebigem Hintergrund, falls dieser für alle Referenzen nahezu konstant ist. Durch die daraus resultierende niedrige Variabilität des Hintergrundes wird dessen Repräsentation bei der Konstruktion des Eigenraums vernachlässigt.

3.5 Ergebnisse und Diskussion

Durch den gewählten Multiskalenansatz in Verbindung mit einem sliding window ergeben sich verschiedene Systemparameter, die zu wählen sind:

- Die Größe des sliding window.
- Die Anzahl zu verwendender Skalierungsstufen beim Multiskalenansatz.
- Schwellenwerte für die verwendbaren Rejectors.

Das sliding window ist stets so groß wie die zur Klassifikation verwendeten Referenzbilder. Die Skalierungsstufen umfassen stets den Bereich zwischen maximaler Verkleinerung der Testszene (so daß noch Bilder in Größe des sliding window ausgeschnitten werden können) und der Originalgröße der Testszene (d.h. die Grösse der Referenzbilder ist gleichzeitig die minimale Größe von Objekten, die gefunden werden können). Demnach kann die Größe eines Objekts in der Szene von minimal Referenzengröße bis maximal Szenengröße variieren. Die Generierung von S Skalierungsstufen erfolgt dann durch Ziehen der S-ten Wurzel aus dem maximalen Verkleinerungsfaktor, was den Skalierungsfaktor zwischen zwei benachbarten Ebenen der Multiskalenpyramide ergibt.
Die benötigte Rechenzeit für das Durchsuchen eines Bildes mit einem sliding window hängt wesentlich von dessen Seitenlänge ab, weshalb in der vorliegenden Arbeit auf COIL-20 nicht mit einem Fenster der Grösse 128x128 gearbeitet wird, sondern die die Referenzen verkleinert werden. Dabei ist zu beachten, daß die Verkleinerung so gewählt wird, daß die Klassen für den Klassifikator separierbar bleiben. Die Verkleinerung der Referenzen bedeutet ferner (durch die Unterabtastung) eine gewisse Toleranz gegenüber leichten Bildstörungen wie geringes hochfrequentes Rauschen und vereinzelte Pixelfehler. In der Arbeit von Theiner [36] können die verwendeten Bilddaten (Radiographien) beispielsweise sehr stark (ca. Faktor 50) verkleinert werden, ohne daß es zu einem signifikaten Abfall der Erkennungsleistung kommt.

Auf dem USPS-Korpus, zu dem aus den Testdaten künstlich Testszenen erzeugt wurden, stellt sich diese Frage nicht: Es wird mit einem 16x16 Pixel großen sliding window gearbeitet. Anhand der

Experimente mit USPS sollen einige Einstellungen (und Probleme) des Bildindizierers motiviert und erläutert werden, die danach bei Experimenten auf COIL-20 überprüft werden.

3.5.1 Experimente mit USPS

Aufgrund der anfallenden Rechenzeiten und der begrenzten Zeit für die vorliegende Arbeit beschränken sich Experimente mit Daten aus dem USPS-Korpus auf die Benutzung der ersten 100 Testbilder. Die 1-Nearest-Neighbour-Fehlerrate für die Einzelobjekterkennung liegt hier bei 6%. Für Experimente mit dem Bildindizierer werden die Testbilder zufallsgesteuert auf eine Grösse zwischen 16x16 und 32x32 Pixel skaliert und an zufällig ausgewählte Positionen in ein Bild der Grösse 96x96 Pixel (mit schwarzem Hintergrund) eingesetzt. Für die Multiskalenpyramide werden 10 Stufen gewählt, die oberste Stufe liegt bei 16x16 Pixeln (Skalierungsfaktor 0.167, 1 Merkmalsvektor), die unterste bei 96x96 Pixeln (Skalierungsfaktor 1.0, 81x81 Positionen, d.h. 6561 Merkmalsvektoren).

Nearest Neighbour

In einem ersten Experiment wird der ‚normale' 1-Nearest-Neighbour-Klassifikator benutzt, ohne daß Mechanismen zum Verwerfen aussichtsloser Bildregionen oder die Handicap-Distanz zum Einsatz kommen. Dementsprechend wird die Multiskalenpyramide vollständig durchsucht, was 1+25+81+196+400+784+1444+2401+4096+6561 = 15989 Merkmalsvektoren pro Tetsszene ergibt. Es wird eine Fehlerrate von 73% erzielt. Abbildung 3.13 stellt die Probleme dar, die durch das alleinige Betrachten eines Ausschnitts der Szene entstehen. Die Referenzen, hier die klassenbesten nächsten Nachbarn, (ursprünglich 16x16 Pixel groß) sind so skaliert und positioniert, daß sie mit dem Bildbereich der Testszene, den sie erklären, korrespondieren. Zum besseren Verständnis sind die Bildregionen, welche durch die Referenzen erklärt werden, mit einem Rahmen markiert. Man beachte, daß der Vergleich stets in einem 16x16 Pixel großen Fenster stattfindet, die Testszene also beispielsweise für den Treffer ‚9', der ziemlich weit oben in der Multiskalenpyramide erzielt wurde, stark verkleinert ist. Die beiden Referenzen mit den geringsten Distanzen ‚erklären' nur einen sehr kleinen Teil der Szene (d.h. der Ziffer ‚0'), sie wurden auf der untersten Stufe der Multiskalenpyramide erzielt. Die fehlerhaft gefundene Ziffer ‚2' stellt ferner ein Problem dar, weil die Grauwertmasse dieser Referenz sehr nah an Null liegt, d.h. es treten zu Bildausschnitten, die ebenfalls sehr wenig Vordergrundinformation enthalten (etwa eine stark verkleinerte Testszene, siehe Abbildung 3.14), nur geringe Distanzen auf. Der dritte Treffer erklärt zwar alle Vordergrundpixel der Testszene, aber als zu kleinen Teil einer großen Referenz. Im Vergleich zu der korrekten Antwort, die erst an vierter Stelle (aller Klassen) genannt wird, geht durch die Skalierung sehr viel weniger Grauwertmasse der zu erklärenden Beobachtung in den Vergleich zwischen der Referenz ‚9' und ‚0' ein als in den Vergleich zwischen der Referenz ‚0' und ‚0''

Nearest Neighbour mit vorgeschaltetem Rejector

Als nächstes werden alle Bildpositionen verworfen, deren Region bezüglich Entropie und Grauwertmasse (gemessen als Summe der Grauwertquadrate) den Wertebereich der Trainingsdaten

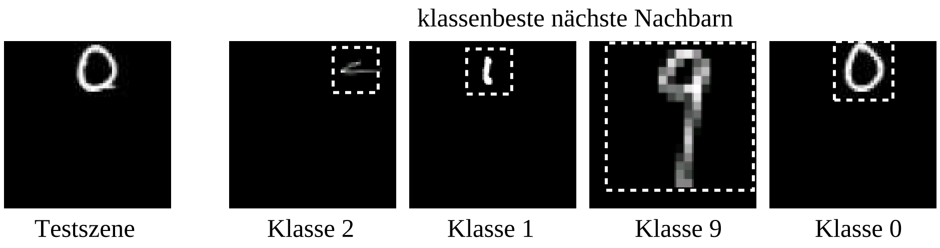

Abbildung 3.13: 2 typische Fehler rein lokaler Entscheidungen bei der Bildindizierung: Die Fehler treten auf, da zu wenig Bildinformation in die Entscheidung eingeht: Die ersten beiden Treffer erklären nur einen geringen Teil der Bildinformation, der dritte Treffer ‚9' erklärt die durch Verkleinerung zu stark reduzierte Bildinformation. Beides resultiert in einer ungenügenden Erkennungsleistung.

plus einem Toleranzbereich von ± 10% verlassen. Die Fehlerrate sinkt hierdurch auf 58% und die Anzahl zu klassifizierender Vektoren wird auf ca. 3000-4000 pro Bild gesenkt.

Eine weitere Möglichkeit ist das Verwerfen von Bildausschnitten, bei denen ein zu hoher Anteil der im Bild vorhandenen Grauwertmasse nicht überdeckt wird. Hierbei wird allerdings das Wissen benutzt, daß sich in jeder Testszene lediglich eine Ziffer befindet. In diesem Experiment wird der Schwellenwert sehr restriktiv eingesetzt: Als Einstellung für den Handicap Rejector wird eine Schwelle von 0:0.1 verwendet, d.h. alle Bildausschnitte, für welche die außenliegende Grauwertmasse mehr als 10% der innenliegenden Grauwertmaße beträgt, werden verworfen. Dementsprechend liegt die Anzahl von pro Testszene untersuchten Merkmalsvektoren bei ca. 200 bis 300. Die erzielte Fehlerrate liegt bei 45%. In Abbildung 3.14 ist ein typisches Problem dargestellt, welches auch durch die Verwendung des Handicap-Rejectors nicht behoben werden kann: Das Handicap verhindert lediglich, daß zu kleine Ausschnitte der Testszene als Treffer akzeptiert werden (also beispielsweise die beiden ersten Treffer in Abbildung 3.13). Die auftretenden Fehler resultieren aus dem umgekehrten Fall: Die Referenz erklärt einen zu großen Bildbereich (welcher ein Handicap von Null besitzt).

Nearest Neighbour mit Handicap-Distanz

Da nur ein einzelnes Objekt pro Testszene vorliegt und diese über homogenen, schwarzen Hintergrund verfügt, wird in diesem Experiment statt der Distanz zwischen Bildausschnitt und Referenz nun die Distanz des Bildausschnitts zu den Referenzen mit fortgesetztem Hintergrund betrachtet. Damit sollten Fehler wie der durch die beiden ersten Treffer in Abbildung 3.13 vermieden werden. Es wird eine Fehlerrate von 9% erzielt. Im Vergleich zur Anwendung des Handicap-Rejectors erfolgt bei Anwendung der Handicap-Distanz eine Normierung der auftretenden Distanzen auf die Anzahl der Pixel der betrachteten Skalierungsstufe, wodurch falschen Treffer wie die in Abbildung 3.14 deutlich herabgestuft werden.

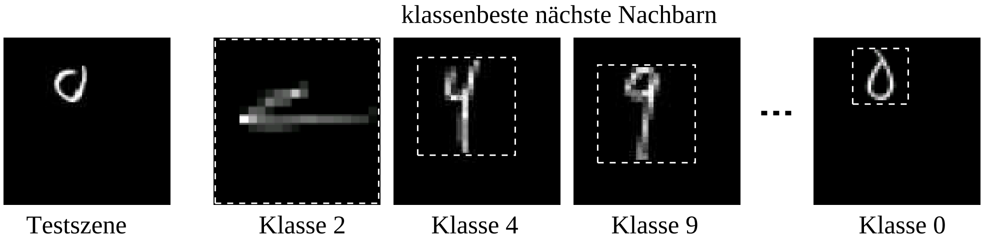

Abbildung 3.14: Ein Fehler bei der Bildindizierung, der vom Handicap-Rejector nicht verhindert werden kann. Das Handicap (d.h. die Benutzung der Grauwertmasse ausserhalb des aktuell betrachteten Bildausschnitts für die Distanzberechnung oder für einen Rejector) ist für alle 3 falschen Treffer Null. Durch die geringe Grauwertmasse der Referenzen entstehen geringere Distanzen zu der stark verkleinerten Testszene als für die korrekte Klasse auf der korrekten Skalierungsstufe.

Nearest Neighbour mit Handicap-Distanz und vorgeschaltetem Rejector

Ein kombinierter Rejector basierend auf Handicap (Schwelle 1:0.1) und Trainingsdaten-Wertebereichen ($\pm$10% Entropie und Grauwertmasse) erzielt bei Verwendung der Handicap-Distanz schließlich eine Fehlerrate von 6%, d.h. ein ebensogutes Ergebnis wie der 1-NN auf dem USPS-Standardkorpus. Eine Zusammenfassung der erzielten Ergebnisse ist in Tabelle 3.1 dargestellt.

Tabelle 3.1: Bildindizierungs-Fehlerrate auf Szenen mit den ersten 100 USPS-Testziffern in Abhängigkeit verschiedener Maßnahmen. Keine Helligkeitsanpassung, Multiskalenpyramide mit 10 Stufen.

Vorgehen	Fehlerrate [%]
lokaler Nearest Neighbour	73
lokaler Nearest Neighbour, Wertebereich-Rejector	58
lokaler Nearest Neighbour, Handicap-Rejector	45
Handicap Nearest Neighbour	9
Handicap Nearest Neighbour, kombinierter Rejector	6

In gewisser Weise und vorausgesetzt, man trifft die korrekte Skalierung, entspricht der Einsatz des sliding windows der in Abschnitt 2.6.2 vorgestellten *Virtual Test Sample*-Methode mit Benutzung der Maximum-Regel. Betrachtet man beim Auswerten der Karte die Nachbarschaft einer Bildposition, lassen sich auch die übrigen VTS-Regeln benutzen.

Multi-Objekterkennung

Zur Multi-Objekterkennung wurden in Anbetracht der gegebenen Zeit nur einige kurze Experimente durchgeführt. Die Schwierigkeit der Aufgabe besteht darin, daß neben der Klassenzugehörigkeit und Position der Objekte in der Testszene auch die Anzahl an Objekten unbekannt ist. Das Erkennungssystem muß also feststellen, ob keine (weiteren) Objekte mehr in einer Szene vorhanden sind.

Ferner sind gewisse Mechanismen, welche die Fehlerrate (und die Laufzeit) bei der Indizierung von Szenen mit nur einem Objekt hilfreich sind, nun nicht mehr anwendbar, beispielsweise der Einsatz des globalen Handicaps für einen Ausschnitt. Für Experimente auf USPS wurden Ziffern aus dem USPS-Testkorpus in variabler Anzahl (1 bis 4) und Skalierung (16x16 bis 32x32) in Testszenen eingesetzt, welche wie bei der Einzelobjekt-Erkennung schwarzen Hintergrund aufweisen.
Die Berechnung des Handicap (siehe Abschnitt 3.4.4, (3.17)) erfolgt hier -aus bereits erwähnten Gründen- lediglich in einer lokalen Umgebung um jeden Bildausschnitt. Abbildung 3.15 zeigt die Resultate exemplarisch durchgeführter Indizierungen auf einigen Beispielbildern. In dem Experiment wurde mit einem lokalen Handicap gearbeitet, welches eine Umgebung mit der dreifachen Fläche des aktuellen Bildausschnitts umfaßt (d.h. bei einem 24x24 Pixel großen sliding window umfaßt das Handicap eine Fläche von 48x48 Pixeln, in deren Mitte der Bildausschnitt liegt). Als Handicap-Rejector-Schwelle wird 1:1 verwendet und der Rejector für Trainingsdaten-Wertebereichsprüfung wird benutzt. Die Handicap-Distanz wurde nicht verwendet, stattdessen findet pro Bildausschnitt Energienormierung statt. Ohne Energienormierung und Verwendung der Handicap-Distanz entsteht ein vergleichbares Ergebnis.

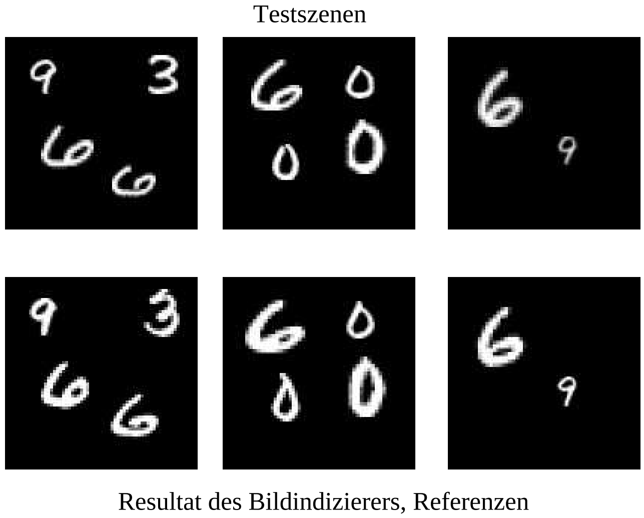

Abbildung 3.15: Multi-Objekterkennung auf USPS. Man beachte die sehr konservative Anordnung der Testobjekte. Für sehr eng beieinanderliegende Ziffern sind Mechanismen wie das Handicap für Distanzberechnung und reject nur sehr eingeschränkt nutzbar. Oben: Testszenen, unten: Zu erkannten Objekten wurde die Referenz zum Objekt korrespondierend skaliert und an die entsprechende Position gesetzt.

3.5.2 Experimente mit COIL-20

Bei den Experimenten wird der mit *processed* bezeichnete Teil der COIL-20-Datensammlung als Referenzdatenmenge benutzt und der mit *unprocessed* bezeichnete (ohne Modifikationen) als Testkorpus. Um sicherzustellen, daß Trainings- und Testdatenmenge wirklich unterschiedlich sind (aus [24] geht nicht eindeutig hervor, ob der *processed*-Korpus direkt aus dem *unprocessed*-Korpus gewonnen wurde), werden jeweils 36 Bilder (die zu ungeraden Drehwinkeln) jeder Klasse aus dem

processed-Korpus als Referenzen und die Bilder zu den jeweils übrigen 36 (d.h. geraden) Drehwinkeln aus dem *unprocessed*-Korpus als Testszenen verwendet. Wie bereits angesprochen, wird nicht mit den 128x128 Pixel großen Referenzbildern gearbeitet, sondern diese werden herunterskaliert. Hierbei wird mit verschiedenen Größen experimentiert. Ebenso soll untersucht werden, wie feinstufig die Multiskalenpyramide aufgebaut werden muß, um die Sensitivität des euklidischen Distanzmaßes gegenüber Skalierungsunterschieden (siehe Abschnitt 2.6.3) zu kompensieren. Zu beachten ist, daß obwohl nur Testdaten aus 5 Klassen vorliegen, die Bildindizierung stets als 20-Klassen-Problem behandelt wird.
In einem ersten Experiment wird der auf USPS (siehe entsprechender Abschnitt) erfolgreich verwendete kombinierte Rejector benutzt, d.h.

- Verwerfen von Bildausschnitten, die den Wertebereich der Referenzen für Grauwertmasse und Entropie um mehr als eine Toleranz von 10% verlassen.

- Verwerfen von Bildausschnitten, deren Handicap höher als 1:1 ist, d.h. falls die Grauwertmasse außerhalb des Bildausschnitts höher als die innerhalb ist.

Für die Multiskalenpyramide werden zunächst 15 Stufen gewählt. Die Ergebnisse in Abhängigkeit der verwendeten Größe des sliding windows, d.h. der Größe der zur Klassifikation verwendeten Referenzbilder, sind in Tabelle 3.2 dargestellt.

Tabelle 3.2: Bildindizierungs-Fehlerrate auf COIL-20 in Abhängigkeit der Größe des sliding windows, keine Helligkeitsanpassung, Multiskalenpyramide mit 15 Stufen, kombinierter Rejector (Handicap-Reject-Schwelle 1:1, Referenzen-Wertebereich ±10% für Entropie, Grauwertmasse)

Größe des sliding window	Fehlklassifikationen (von 180)
16x16	0+12+3+9+0 = 13.3%
24x24	0+7+6+11+0 = 13.3%
32x32	0+9+6+10+0 = 13.9%

Die Helligkeitsabweichung zwischen Referenzen und den Objekten in den Testszenen ist offenbar zu hoch, um eine gute Erkennung zu ermöglichen. Deshalb wird in einem nächsten Experiment die Energienormierung benutzt, um Referenzen und ausgeschnittene Bildbereiche anzupassen. Die hierbei erzielten Ergebnisse sind in Tabelle 3.3 dargestellt.
Offenbar scheint eine Fenstergröße (und damit Referenzengröße) von ca. 24x24 Pixeln einen geeigneten Kompromiß zwischen Detailauflösung (zur Diskriminanz zwischen Klassen) und Unterabtastung (zur Robustheit gegenüber geringen lokalen Störungen und zur Beschleunigung des Erkennungsvorgangs) zu sein. In Abbildung 3.16 sind Beispiele zu Antworten des Bildindizierers dargestellt, die auch einen Eindruck von der Auflösung geben, auf der Beobachtung (d.h. Inhalt des sliding windows) und Referenzbilder verglichen werden.
Ferner soll untersucht werden, inwieweit die Auflösung der Multiskalenpyramide Einfluß auf die Erkennungsleistung nimmt. Ergebnisse hierzu sind in Tabelle 3.4 dargestellt.

Tabelle 3.3: Bildindizierungs-Fehlerrate auf COIL-20 in Abhängigkeit der Größe des sliding windows, Helligkeitsanpassung durch Energienormierung, Multiskalenpyramide mit 15 Stufen, kombinierter Rejector (Handicap-Reject-Schwelle 1:1, Referenzen-Wertebereich ±10% für Entropie, Grauwertmasse)

Größe des sliding window	Fehlklassifikationen (von 180)
16x16	0+8+5+5+0 = 10.0%
24x24	0+1+3+0+0 = 2.2%
32x32	0+4+7+6+0 = 9.4%

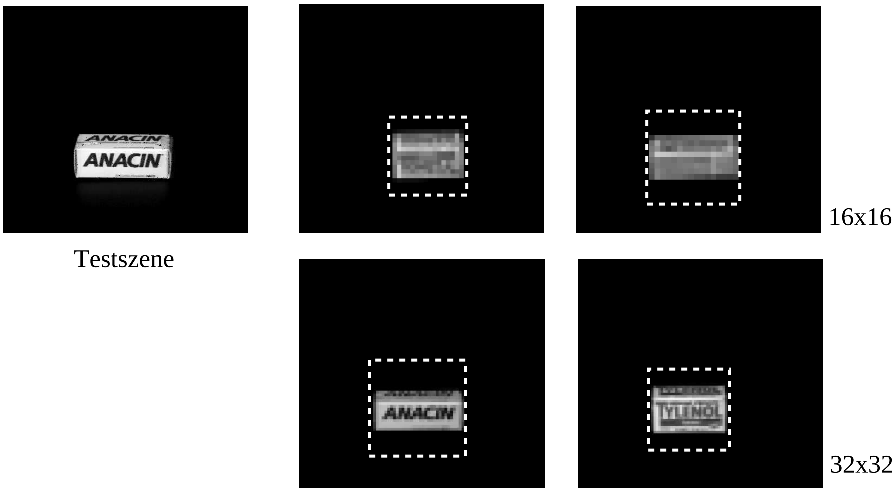

Abbildung 3.16: Einige Antworten des Bildindizierers für COIL-Testszenen. Gut zu erkennen ist die unterschiedliche Helligkeit bei Testszenen (linke Spalte) und Referenzen (übrige Spalten, klassenbeste Treffer, in von links nach rechts fallender Wahrscheinlichkeit). Oben: 16x16 Pixel großes sliding window: Die niedrige Auflösung des sliding windows (und damit der Referenzbilder) erschwert die korrekte Klassifikation einer Schachtel der Klasse 5 unten: 32x32 Pixel großes sliding window: Nun sind die Schachteln besser unterscheidbar. (vergleiche Abbildung 3.2).

Wie erwartet, ermöglicht eine eine feinere Auflösung der Skalierungsstufen eine bessere Erkennungsleistung. Die Ergebnisse zeigen, wie wichtig das ‚Treffen‘ der richtigen Skalierungsstufe für eine akzeptable Erkennungsleistung ist, insbesondere bei Verwendung des Euklidischen Abstandes als Distanzmaß. Die fehlerfreie Indizierung der Testklassen 13 und 7 (Testklasse 1 und 5) bei Benutzung von lediglich 5 Skalierungsstufen ist möglich, da hier zufällig die Skalierung der Testobjekte getroffen wird. Bei Testszenen der anderen Klassen ist dies offensichtlich nicht der Fall.
Für 40 Skalierungsstufen ist eine fehlerfreie Indizierung aller Testszenen möglich. Hierbei werden pro Bild etwa 5000-6000 Merkmalsvektoren klassifiziert (der Rest wird durch den kombinierten Rejector verworfen).

Des weiteren soll der Einfluß des Schwellenwertes für den Handicap-Rejector untersucht werden,

Tabelle 3.4: Bildindizierungs-Fehlerrate auf COIL-20 in Abhängigkeit der Auflösung der Multiskalalenpyramide. Helligkeitsanpassung (durch Energienormierung), Fenstergröße 24x24 kombinierter Rejector (Handicap-Reject-Schwelle 1:1, Referenzen-Wertebereich ±10% für Entropie, Grauwertmasse)

Anzahl Skalierungsstufen	Fehlklassifikationen (von 180)
5	0+22+22+36+0 = 44.4%
10	0+4+1+6+0 = 6.1%
15	0+1+3+0+0 = 2.2%
20	0+0+1+2+0 = 1.7%
30	0+0+0+2+0 = 1.1%
40	0+0+0+0+0 = 0%

d.h. inwieweit sich die Erkennungsleistung ändert, falls Bildbereiche als Objektpositionen zugelassen werden, welche bei einer möglichen Segmentierung eher unwahrscheinlich wären. Die Ergebnisse zu diesem Experiment sind in Tabelle 3.5 dargestellt. Anscheinend ergeben sich auf COIL nicht derart viele Fehlerquellen zur falschen Erkennung eines Szenenteilbereichs als Objekt wie auf USPS, so daß dieser Parameter auf COIL weniger wichtig ist.

Tabelle 3.5: Bildindizierungs-Fehlerrate auf COIL-20 in Abhängigkeit der Handicap-Reject-Schwelle. Helligkeitsanpassung (durch Energienormierung), Fenstergröße 24x24, 15 Skalierungsstufen kombinierter Rejector (Handicap-Reject-Schwelle 1:1, Referenzen-Wertebereich ±10% für Entropie, Grauwertmasse)

Handicap-Reject-Schwelle	Fehlklassifikationen (von 180)
1:0.1	0+0+2+0+0 = 1.1%
1:1	0+1+3+0+0 = 2.2%
1:2	0+1+1+2+0 = 2.2%

Ergebnisse für andere Aufteilungen des COIL-20 Korpus

Im Rahmen weiterer Untersuchungen wird eine Indizierung aller Bilder des Testkorpus von COIL-20 (*unprocessed*) unter Verwendung aller Bilder des Trainingskorpus (*processed*) als Referenzen durchgeführt. Dabei wird der kombinierte Rejector für Handicap (1:0.1) und Referenzen-Wertebereich (±10% bezüglich Entropie und Grauwertmasse) verwendet. Hierbei ergibt sich für eine sliding-window-Größe von 28x28 Pixel und die Wahl von 15 Skalierungsstufen eine Fehlerrate von 0%.

Untersuchungen im Rahmen normaler Objektklassifikation, wie z.B. die Arbeiten von Baker und Nayar [1] oder die Arbeit von Schiele [30] benutzen eine Aufteilung des COIL-20 Trainingskorpus *processed* in eine Trainings- und Testdatenmenge. Hier erzielt allerdings bereits der 1-NN-Klassifikator für Bildgrößen von z.B. 16x16, 24x24 und 32x32 eine Fehlerrate von 0%.

Versuche mit inhomogenem Hintergrund

Pösl und Niemann evaluieren das in [27] vorgestellte System u.a. anhand manuell erstellter Testszenen. Der *processed*-Korpus von COIL-20 wird hier für in 2 disjunkte Mengen von Referenz- und Testbildern aufgeteilt und die Testbilder jeweils einzeln in eine Testszene mit inhomogenem Hintergrund eingesetzt. Dokumentierte Versuche wurden lediglich mit der Klasse ‚Sparschwein' (siehe Abbildung 3.2) durchgeführt, d.h. es findet lediglich die Lokalisierung statt, Objektklasse und Skalierung sind fest. Um die in Abschnitt 3.4.6 vorgestellte Behandlung von inhomogenem Hintergrund zu evaluieren, wurde ein ähnlicher Versuchsaufbau gewählt. Allerdings wird von dem in der vorliegenden Arbeit beschriebenen System nicht den Parameterraum für Objektrotationen in der Bildebene abgesucht, sondern lediglich den Parameterraum der Translation. Andererseits wird, da der Schwerpunkt der vorliegenden Arbeit auf der Klassifikation im Bild vorhandener Objekte abzielt, in diesem Experiment nach Objekten aus allen 20 Klassen in der jeweiligen Testszene gesucht, d.h. es wird mit 720 ($= 36 \cdot 20$) Referenzbildern gearbeitet. Die 36 Testbilder der Klasse ‚Sparschwein' werden zufällig in eine Testszene mit inhomogenem Hintergrund eingesetzt. Um analog zu Pösl und Niemann mit einer festen Skalierung der Objekte zu arbeiten, werden 100x100 Pixel große Testszenen verwendet, in die Beobachtungen der Größe 24x24 Pixel eingesetzt sind. Die Indizierung findet nur auf dieser Skalierungsstufe statt. Das Handicap wird aufgrund des inhomogenen Hintergrundes nicht verwendet, ebenso werden die Trainingsdaten-Wertebereich-Rejectors nicht verwendet. Somit werden alle $77 \cdot 77 = 5929$ Positionen im Bild abgesucht. Bei der Distanzberechnung werden keine Distanzen an Pixeln berechnet, bei denen die jeweilige Referenz Hintergrund aufweist. Der Hintergrund in den Referenzen wurde mit einem schwellenwertgesteuerten flood-fill bestimmt. Die Rechenzeit auf einer Digital Alpha-Maschine bewegt sich für 720 Referenzen und 5929 Bildausschnitte bereits im Bereich von 2-3 Minuten. Im Experiment wurden 30 der 36 Testszenen korrekt indiziert, d.h. das Sparschwein wurde jeweils als bester Treffer und in korrekter Lokalisierung zurückgeliefert. In einem der Fehlerfälle befindet sich das Sparschwein nicht unter den besten 3 Klassen (in den übrigen 5 Fällen ist die Klasse Sparschwein mit korrekter Lokalisierung stets an zweiter Position).

3.5.3 Diskussion der Ergebnisse und aufgetretener Probleme

Bei den auf USPS für 100 Testdaten expemplarisch durchgeführten Untersuchungen konnte die bei der Einzelobjekterkennung (siehe Kapitel 2) erzielte Fehlerrate auch für beliebige Skalierungen und Translationen der Testdaten in einer Szene mit homogenem Hintergrund erreicht werden, ohne daß auf Segmentierung zurückgegriffen wurde. Aufgrund zahlreicher Fehlerquellen wie Verwechslungsmöglichkeit von Ziffern durch nur leicht falsche Lokalisation, z.T. ungünstiges Trainingsmaterial, welches sehr wenig Vordergrundinformation enthält und die geringe ‚Flächigkeit' der Daten (wodurch die Distanzen sehr sensibel gegenüber schon geringen Transformationen des Bildmaterials sind) gestaltet sich die Erkennung der Objekte recht schwierig. Ferner sind einige der vorgestellten Mechanismen nur unter Nebenbedingungen (homogener Hintergrund, Einzelobjektsuche) anwendbar, was zu eingeschränkter Leistungsfähigkeit des Systems bei der Multi-Objekterkennung führt. Bei einem (nur exemplarischen) Experiment erwies sich das lokale Handicap als reject-Kriterium brauchbar und das Resultat des Systems gut.

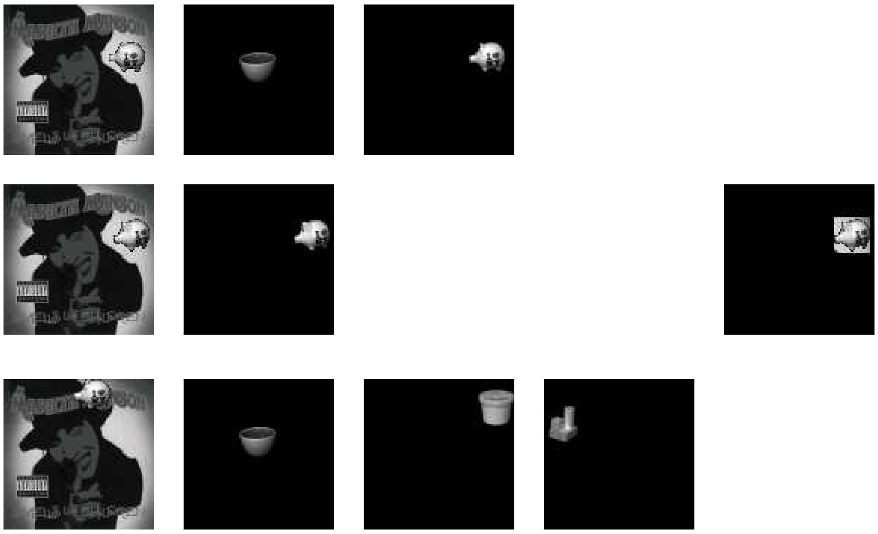

Abbildung 3.17: Objektsuche bei inhomogenem Hintergrund. Die Skalierung ist fest, Trainings- und Testdaten sind zwar disjunkt, aber beide aus dem *processed*-Korpus von COIL-20 entnommen. Erste Zeile: Eine der 6 Fehlklassifikationen, das korrekte Objekt ist an zweiter Stelle der klassenbesten Nachbarn. Zweite Zeile: Korrekte Lokalisierung und Klassifikation (eine von 30), ganz rechts der Bildausschnitt, für den der Treffer erzielt wurde. Dritte Zeile: Die Klasse Sparschwein befindet sich nicht unter den 3 besten Klassen.

Die für COIL-20 erzielten Resultate bestätigen die Eignung des Systems zur Lokalisation und Klassifikation von Einzelobjekten vor homogenem Hintergrund. Beim Vergleich mit dem Ergebnis von Murase und Nayar, die in [22] 0% Fehlerrate erzielen, muß die eingeschränkte Vergleichbarkeit berücksichtigt werden. Die Autoren verwenden den vollen *unprocessed*-Korpus von COIL-20 (1440 Bilder, Referenzen in 5°-Schritten) zur Erkennung von 320 nicht verfügbaren Testszenen mit Objekten aller 20 Klassen, deren Drehwinkel frei gewählt wurden. Dies ergibt eine maximal mögliche Abweichung von 2.5°. Die Experimente in der vorliegenden Arbeit beziehen sich auf den *unprocessed*-Korpus. Zuvor wurden Trainings- und Testkorpus derart modifiziert, daß die Rotationswinkel in beiden Datensätzen disjunkt sind. Dies ergibt eine Auflösung der Trainingsdaten von 10°, und die Testdaten sind stets weitestmöglich von den Referenzen entfernt (5°).
In anderen Arbeiten durchgeführte Trennungen des *processed*-Korpus in eine Trainings- und Testmenge zur einfachen Objekterkennung (ohne Lokalisation, mit fester Skalierung) sind relativ uninteressant, da hier bereits durch den 1-Nearest-Neighbour (für Bildgrößen 16x16, 24x24, 32x32) eine Fehlerrate von Null erzielt wird.
Bei Experimenten mit COIL-20-Objekten vor inhomogenem Hintergrund werden die Grenzen des aktuellen Systems sichtbar, was Erkennungsleistung und Rechenzeitbedarf angeht. Hier sind weitergehende Überlegungen notwendig, wie die Konfidenz in vom System gelieferte Treffer modelliert werden kann.

Kapitel 4

Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde ein statistisches Erkennungssystem vorgestellt, das auf

- erscheinungsbasierter Merkmalsanalyse,
- BAYES'scher Entscheidungsregel,
- statistischer Merkmalsreduktion, sowie
- GAUSS'schen Mischverteilungen zur Modellierung der Datenverteilung im Merkmalsraum

basiert und zunächst ohne problemspezifisches Wissen arbeitet. Das System wurde anhand der USPS-Datensammlung evaluiert. Dabei wurde für die Erkennung handgeschriebener, isolierter Ziffern spezifisches Wissen über die Variabilität des Datenmaterials in das System integriert, und zwar durch

- Trainingsdatenvervielfachung
- VTS, einen Mechanismus zur Kombination von Klassifikatorentscheidungen für vervielfachte Testdaten

Die auf der Datensammlung USPS erzielten Ergebnisse haben sich als konkurrenzfähig zu anderen Ansätzen erwiesen (Support-Vektor-Maschine mit virtuellen Support-Vektoren: 3.2%, vorliegende Arbeit: 3.4%, wobei die Abweichung statistisch nicht signifikant ist). In Rahmen der Vervollständigung der Ergebnisse sind noch Experimente mit GAUSS'schen Mischverteilungen durchzuführen, die volle Kovarianzmatrizen benutzen.
Am i6 ausgeführte Erweiterungen des beschriebenen Erkennungssystems um ein invariantes Distanzmaß bzw. invariante Parameterschätzung [7, 14] schließen die Lücke zu Tangentendistanz-basierten Verfahren und erzielen Fehlerraten von 2.7% (für GAUSS'sche Mischverteilungen) bzw. 2.2% (für Kernel Densities).

Im zweiten Teil der Arbeit wurde untersucht, inwieweit sich Verfahren zur Klassifikation von Einzelobjekten bei Verwendung eines sliding-window-Konzepts und eines Multiskalenansatzes zur

Konstruktion eines Bildindizierers eignen. Dieser sollte segmentierungsfrei Szenen nach darin vorhandenen Objekten durchsuchen, also neben dem Klassifikations- auch ein Lokalisierungsproblem lösen. Hierbei hat sich herausgestellt, daß eine rein lokale Sichtweise eines Bildausschnitts nicht ausreicht, um akzeptable Ergebnisse zu erhalten. Es wurden verschiedene Mechanismen vorgestellt, um die Konfidenz in die lokale Entscheidung des Klassifikators anhand globaler Informationen über die Szene zu quantifizieren, hauptsächlich unter der Nebenbedingung, daß homogener Hintergrund vorliegt. Dazu gehört das Distanz-Handicap, welches eine Referenz als global oder lokal in einer Umgebung des aktuell betrachteten Ausschnitts fortgesetzt betrachtet. Ausgehend von der Idee des Rejectors bzw. einer Art Filterung wurden einige Regeln angewendet, um bestimmte Bildbereiche als ‚uninteressant' einstufen zu können. Für die Datensammlung COIL-20 wurde hierbei ein zu der Arbeit [22] von Murase und Nayar konkurrenzfähiges Resultat von 0% Fehlerrate erzielt. Für die Erkennung von Objekten vor beliebigem Hintergrund und die Erkennung mehrerer Objekte sind weitergehende Überlegungen durchzuführen. Die Konfidenz für Treffer könnte mit der Übertragung der VTS-Methode auf den Bildindizierer besser abgeschätzt werden. Es sollte untersucht werden, ob und wie sich globale Informationen über die Szene bzw. die Nachbarschaft des aktuell betrachteten Ausschnitts in Verbindung mit Merkmalsreduktion benutzen lassen. Ferner interessant ist eine eingehendere Untersuchung verwendbarer Distanzmaße bzw. die Formulierung von Regeln und Maßen, welche helfen, interessante Regionen (engl. *roi, region of interest*) in einer Szene zu finden, um einerseits die Qualität der Erkennung zu steigern und andererseits die mit dem sliding-window-Ansatz verbundenen hohen Rechenzeiten zu senken. Ein Ansatz zur Beschleunigung der Suche von Korrespondenzen zwischen Referenzen und Bildregionen einer Testszene ergibt sich aus dem Faltungstheorem [13]: Durch Produktbildung zweier Fourier-transformierter Bilder ergibt sich nach der Rücktransformation des Produktbildes eine Karte, auf der Punkte hoher Korrelation zwischen Referenz und Szene ablesbar sind. Zur Verbesserung der Erkennungsleistung können weitere Merkmale wie Texturinformationen, Kanteninformationen oder ähnliches herangezogen werden, die weitere Quellen zur Formulierung von Konfidenz in Treffer oder die Bestimmung interessanter Bildbereiche sind.

Literaturverzeichnis

[1] S. Baker, S. K. Nayar: *Pattern Rejection*, Proceedings of Computer Vision and Pattern Recognition, San Francisco, 1996.

[2] L. Bottou, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, L. Jackel, Y. Le Cun, U. Müller, E. Säckinger, P. Simard, V.N. Vapnik: *Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition*, Proceedings of the International Conference on Pattern Recognition, Jerusalem, Israel, IEEE Computer Society Press, 1994.

[3] C.J.C. Burges: *A Tutorial on Support Vector Machines for Pattern Recognition*, Data Mining and Knowledge Discovery, Vol.2, No.2, 1998.

[4] C. Cortes, V. Vapnik: *Support Vector Networks*, Machine Learning 20, 1995.

[5] J. Dahmen, K. Beulen, M.O. Güld, H. Ney: *A Mixture Density Approach to Object Recognition for Image Retrieval*, 6th International RIAO Conference on Content-Based Multimedia Information Access, Seite 1632-1647, Paris, Frankreich, April 2000.

[6] J. Dahmen, J. Hektor, R. Perrey, H. Ney: *Automatic Classification of Red Blood Cells using Gaussian Mixture Densities*, Bildverarbeitung für die Medizin, München, Seite 331-335, März 2000.

[7] J. Dahmen, D. Keysers, M.O. Güld, H. Ney: *Invariant Image Object Recognition*, 15th International Conference on Pattern Recognition, Barcelona, Spanien, September 2000.

[8] J. Dahmen, D. Keysers, M. Pitz, H. Ney: *Structured Covariance Matrices for Statistical Image Object Recognition* 22. DAGM Symposium Mustererkennung, Kiel, September 2000.

[9] J. Dahmen, T. Theiner, D. Keysers, H. Ney, T. Lehmann, B. Wein: *Classification of Radiographs in the 'Image Retrieval in Medical Applications'-System (IRMA)* 6th International RIAO Conference on Content-Based Multimedia Information Access, Seite 551-566, Paris, Frankreich, April 2000.

[10] A. Dempster, N. Laird, D. Rubin: *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society, 39(B), Seite 1-38, 1977.

[11] R. Duda, P. Hart: *Pattern Classification and Scene Analysis*, Wiley & Sons, 1973.

[12] T. Hastie, R. Tibshirani, A. Buja: *Flexible Discriminant and Mixture Models* in: J. Kay, D. Titterington (Herausgeber): Proceedings of Neural Networks an Statistics conference, Oxford University Press, Edinburgh, Seite 1-23, 1995.

[13] Bernd Jähne: Digitale Bildverarbeitung, 4. Auflage, Springer 1997.

[14] D. Keysers: *Approaches to Invariant Image Object Recognition*, Diplomarbeit am Lehrstuhl für Informatik VI, RWTH Aachen, Juni 2000.

[15] J. Kittler, M. Hatef, R.P.W. Duin, J. Matas: *On Combining Classifiers*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.20, No.3, März 1998.

[16] Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel: *Handwritten digit recognition with a back-propagation network*, Advances in Neural Information Processing Systems, 1990.

[17] Y. Le Cun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J.S. Denker, H. Drucker, I. Guyon, U. Müller, E. Säckinger, P. Simard, V. Vapnik: *Comparison of Learining Algorithms for Handwritten Digit Recognition*, in: F. Fogelman, P. Gallinari (Herausgeber): International Conference on Artificial Neural Networks, Seite 53-60, Paris, 1995.

[18] T. Lehmann, W. Oberschelp, E. Pelikan, R. Repges: *Bildverarbeitung für die Medizin*, Springer 1997.

[19] Y. Linde, A. Buzo, R. Gray: *An Algorithm for Vector Quantizer Design*, IEEE Transactions on Communications, Vol. 28, No.1, Seite 84-95, 1980.

[20] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, K.-R. Müller: *Fisher Discriminant Analysis with Kernels*, in Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas (Herausgeber): Neural Networks for Signal Processing IX, Seite 41-48. IEEE, 1999.

[21] B. Moghaddam, A. Pentland: *Probabilistic Visual Learning for Object Representation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.19, No.7, Juli 1997.

[22] H. Murase, S. K. Nayar: Visual Learning and recognition of 3-D Objects from Appearance, International Journal of Computer Vision, 14, Seite 5-24, 1995.

[23] D. Nauck, F. Klawonn, R. Kruse: *Neuronale Netze und Fuzzy-Systeme* Vieweg, 2. Auflage 1996.

[24] S. A. Nene, S. K. Nayar, H. Murase: *Columbia Object Image Library (COIL-20)* Technical Report CUCS-005-96, Februar 1996.

[25] H. Ney: *Mustererkennung und Neuronale Netze*, Skript zur Vorlesung, RWTH Aachen, 1999.

[26] H. Ney, L. Welling, S. Ortmanns, K. Beulen, F. Wessel: *The RWTH Large Vocabulary Continuous Speech Recognition System and Spoken Word Retrieval*, Proc. 24th Annual Conference of the IEEE Industrial Electronics Society, S. 2022-2027, September 1998.

[27] J. Pösl, H. Niemann: *Object Localization with Mixture Densities of Wavelet Features*, International Wavelets Conference, Tanger, Marokko, April 1998.

[28] J. Pösl, H. Niemann: *Statistical 3-D Object Localization Without Segmentation Using Wavelet Analysis*, Computer Analysis of Images and Patterns (CAIP), S. 440-447, Kiel, September 1997.

[29] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery: *Numerical Recipes in C*, 2. Auflage, Cambridge University Press, 1992.

[30] B. Schiele: *Object Classification based on Visual Classes*, in E. Paulus, F.M. Wahl: Mustererkennung 1997, Springer 1997.

[31] B. Schiele, J. L. Crowley: *Probabilistic Object Recognition using Multidimensional Receptive Field Histograms*, International Conference on Pattern Recognition 1996, Vol. B, Seite 50-54, August 1996.

[32] B. Schölkopf: *Support Vector Learning*, Oldenbourg Verlag, 1997.

[33] M. Schreckenberg, J. Dahmen, M.O. Güld, G. Schummers, D. Meyer-Ebrecht, H. Ney: *Automatische Endokarderkennung in 3D mit approximierenden Thin-Plate-Spline Modellen unter Einsatz von Gauß'schen Mischverteilungs-Modellen für die lokale Klassifikation*, Bildverarbeitung für die Medizin 2000, S. 351-355, März 2000.

[34] E. Schukat-Talamazzini: *Automatische Spracherkennung*, Vieweg, 1995.

[35] P. Simard, Y. Le Cun, J.S. Denker: *Efficient Pattern Recognition Using a New Transformation Distance*, Advances in Neural Information Processing Systems, Vol.5, 1993.

[36] T. Theiner: *Inhaltsbasierter Zugriff auf große Bilddatenbanken*, Diplomarbeit am Lehrstuhl für Informatik VI, RWTH Aachen, Februar 2000.

[37] V.Vapnik: *Statistical Learning Theory*, Springer, 1998.

Anhang A

Verwendete Software

Die Implementierung fand unter Linux und Verwendung des GNU C-Compilers (`http://gcc.gnu.org`) statt. Ein weiteres häufig benutztes Werkzeug war der GNU-Debugger (`http://sources.redhat.com/gdb`).
Die Sourcen sowie die Diplomarbeit wurden mit dem vim-Editor verarbeitet (`http://www.vim.org`).
Die Abbildungen entstanden mit Hilfe von tgif (`http://bourbon.cs.umd.edu:8001/tgif`) und GIMP (`http://www.gimp.org`).
Die Diagramme wurden mit gnuplot erstellt (`http://www.gnuplot.org`).

Ferner wurden folgende Bibliotheken benutzt:

- zlib (`ftp://ftp.freesoftware.com/pub/infozip/zlib/zlib.html`)
- gdlib (`http://www.boutell.com/gd`)
- LAPACK (`http://www.netlib.org/lapack/index.html`)
- Singulärwertzerlegung aus [29]

Anhang B

Erstellte Software

B.1 Programme

B.1.1 Hilfsprogramme für Merkmalsvektorfiles

vecfile_concat.c

Fügt mehrere Merkmalsvektorfiles aneinander. Die Files müssen die gleiche Anzahl Klassen und Merkmale besitzen.

vecfile_concat_cmps.c

Fügt Vektoren mehrerer Merkmalsvektorfiles jeweils zu einem Vektor zusammen. Die Files müssen die gleiche Anzahl Klassen und die Klassen müssen pro Vektor in allen Files übereinstimmen.

vecfile_transform_and_reduz.c

Multipliziert die Vektoren eines Merkmalsvektorfiles mit einer Matrix.

B.1.2 Merkmalsreduktion

lda.c

Berechnet die LDA-Transformationsmatrix zu Vektoren eines Merkmalsvektorfiles (siehe Abschnitt 2.5.2).

whitening.c

Berechnet wahlweise

- die Transformationsmatrix zur Diagonalisierung der Kovarianzmatrix von Vektoren eines Merkmalsvektorfiles, (siehe Abschnitt 2.5.1).
- die Transformationsmatrix zum Weißmachen der within-class-scatter-Matrix von Vektoren eines Merkmalsvektorfiles, (siehe Abschnitt 2.5.3).

- die klassenabhängigen Transformationsmatrizen zur Diagonalisierung der Kovarianzmatrix zu jeder Klasse der Merkmalsvektoren eines Merkmalsvektorfiles.

svd.c

Berechnet die Singulärwertzerlegung einer Basis.

train_means_single_global

Berechnet die Vektoren $(\mu_k - \mu), k = 1, \dots, K$ eines Merkmalsvektorfiles.

B.1.3 EM-Training/Klassifikation

train_gmv.c

Implementierung des EM-Trainings für GAUSS'sche Mischverteilungen.

classify_gmv.c

Klassifikationsprogramm für GAUSS'sche Mischverteilungen (siehe Abschnitt 2.4.1).

classify_gmv.vts.c

Klassifikationsprogramm für GAUSS'sche Mischverteilungen, Implementierung der VTS-Methode (siehe Abschnitt 2.6.2).

create_pseudo_classes.c

Clustering-Algorithmus auf Basis GAUSS'scher Mischverteilungen (siehe Abschnitt 2.4.1)

B.1.4 Modellfreie Klassifikatoren

classify_nn.c

Implementierung des Nearest-Neighbour-Klassifikators (siehe Abschnitt 2.4.3)

classify_kd.c

Implementierung des Kernel-Densities-Klassifikators (siehe Abschnitt 2.4.2)

classify_tangent_nn.c

Implementierung des Tangentendistanz-Nearest-Neighbour-Klassifikators (siehe Abschnitt 2.6.3)

classify_eigen_nn.c

Implementierung des DFFS-Nearest-Neighbour-Klassifikators (siehe Abschnitt 2.6.4)

B.1.5 Bildindizierung

create_classify_map.c

Bildindizierer nach Kapitel 3. Erstellt eine Multiscale-Map einer Testszene.

create_classify_map.c

Auswertungsprogramm für Multiscale-Maps (siehe Kapitel 3).

B.2 Dateiformate

Merkmalsvektorfiles

```
Header:
<Anzahl Klassen (int)> <Anzahl Komponenten (int)>

Body:
<Klassennummer (int)> <Vektorkomponente 1 (double)> ... <Vektorkomponente n> (double)>
...

Tail:
-1
```

Transformationsmatrizen auf Basis von Eigenvektoren

```
Header:
<Anzahl Komponenten (= Anzahl Eigenvektoren) (int)>
Body:
<Eigenwert 1 (double)> ... <Eigenwert n (double)>

<Eigenvektor 1, Komponente 1 (double)> ... <Eigenvektor 1, Komponente n (double)>
...
<Eigenvektor n, Komponente 1 (double)> ... <Eigenvektor n, Komponente n (double)>

...

Tail:
-1
```

Mischverteilungsparameter

```
Header:
<Anzahl Klassen (int)> <Anzahl Komponenten (int)>
```

Body:
<Klassenindex k (int)> <Anzahl Dichten Klasse I_k (int)>
<a-priori-Klassenwahrscheinlichkeit Klasse k (double)>
<c_{1k} (double)> <$norm_{1k}$ (double)> ... <$c_{I_k k}$ (double)> <$norm_{I_k k}$ (double)>
...

magic number:
-1

<Klassenindex k (int)>
<mean k, Dichte 1, Komponente 1> <varianz k, Dichte 1, Komponente 1>
...
<mean k, Dichte I_k, Komponente 1> <varianz k, Dichte I_k, Komponente 1>

...

<mean k, Dichte 1, Komponente d> <varianz k, Dichte 1, Komponente d>
...
<mean k, Dichte I_k, Komponente d> <varianz k, Dichte I_k, Komponente d>
...

Tail:
-1

B.3 Skripte

prepare_data.csh

Automatisierter Ablauf der Subspace-Methode zur Merkmalsreduktion.

train_and_classify.csh

Automatisierter Ablauf von Training/Klassifikation für GAUSS'sche Mischverteilungen.

Index