Kapitel 1 Einführung

1.1 Schrifterkennung

Die Rechengeschwindigkeit von Computern nimmt immer mehr zu. Ein Großrechner von 1980 passt heutzutage in eine Tasche. Mit dieser hohen Rechenleistung lassen sich komplete Anwendung realisieren wie z.B. die Handschrifterkennung als eine gute Alternative zur Tastatur.

Oft unterscheidet man sechs Phasen im Handschrifterkennungsprozess. Diese sind in Bild 1 dargestellt.

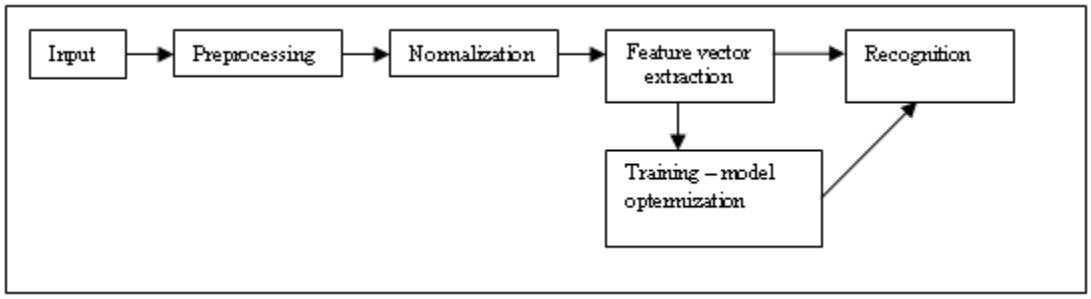


Bild 1 Die sechs Phasen in der Handschrifterkennung Prozess.

Die Eingabe der Information

Man unterscheidet zwischen on-line und off-line Schrifterkennungssoftware. In einem off-line System werden Informationen normalerweise mit einem Scanner eingegeben. In einem on-line System wird die Schriftinformation üblicherweise durch eine Tablett eingegeben. Ein Beispiel für ein on-line Systems ist ein PDA. Ein PDA muss die Schrift innerhalb von Millisekunden erkennen. Ein off-line System hat den Vorteil, dass es keine Zeitvorgabe gibt. Damit kann der Algorithmus komplizierter sein und mehr Zeit für die Ausführung benötigen. Es gibt aber auch einen Nachteil mit einem so genannte off-line System. Die Richtungsinformationen die durch die Führung des Stiftes beim on-line System zur Verfügung stehen, gehen bei einem off-line System verloren, da man nur noch Pixelinformationen besitzt. Der Verlust dieser Information ist aber ein großer Nachteil, da damit einfacher gearbeitet werden könnte als mit den Informationen eines Bildes.

Bild Vorverarbeitung

Es gibt zwei Schritte in der Bildvorverarbeitung. Der erste entfernt Verunreinigungen des Bild. z.B. Störungen die aufgrund von Papierfarbe oder einem Fleck auf dem Scanner verursacht wurden. Der zweite normiert die Handschrift. Die Handschrift wird angepasst. Beispielsweise wird die Neigung der Handschrift verbessert. Es ist üblich das Wort in die Mitte zu schieben. Dazu nutzt man die Schreiblinie. Wenn das Erkennungsystem auf ein zeichenmäßigem System gründet, d.h. jeden Buchstaben getrennt erkennt, dann wird das Wort vorher in einzelne Buchstaben segmentiert.

Die Merkmalsvektoren

Um die Handschrift zu erkennen, müssen Merkmalsvektoren gefunden werden. Dabei gibt es zwei Möglichkeiten. Einerseits kann nach einzigartigen Merkmalen (z.B.

Schleifen, Punkte und Kurven) gesucht werden. Anderseits kann eine Art statistische Analyse gemacht werden, wobei das ganze Bild analysiert wird.

Training

Damit der Rechner die Handschrift „lernen“ kann, werden Neuronale Netzwerke und *Hidden Markov Models* in *state of the art* Systemen verwendet.

Erkennung

Nachdem das Systeme trainiert wurde, sollte es jede Handschrift erkennen können.

Heutzutage existieren viele Arten von Schrifterkennungs – Programmen.
Das gewöhnlichste System heißt OCR (Optical Charakter Recognition), mit dieser Software kann der Benutzer einfach eine A4 Seite auf einen Scanner legen. Diese Seite wird dann in einen ASCII Text übersetzt. Beispiele solcher Systeme sind CalPoly für Linux und IRIS OCR Software für Windows.

Die Grundlagen für dieses Projekt wurden bei M. Dehgahn, K Faez, M. Ahmadi, M Shridhar im Februar 2000 erarbeitet. Sie haben eine Veröffentlichung darüber geschrieben.

1.2 Arabische Handschrift

Arabische Handschrifterkennungssysteme haben Probleme die z.B. lateinische – Handschrifterkennungssysteme (oder auch kyrillische) nicht haben. In Bild 2 steht „Hallo diese Sprache ist deutsch, englisch, arabisch, französisch und russisch“ geschrieben. Wenn man sich z.B. die englische Schrift anschaut ist klar, dass es eine kleine Pause zwischen den Buchstaben gibt. Gleiches stellt man in der der arabischen Schrift fest. Da die arabische Schrift sowohl in handgeschriebener Form, als auch in Druckschrift eine kursive Schrift ist, ist eine Segmentierung in Buchstaben, anders als in z.B englischer Druckschrift, schwer möglich. Mit lateinischen Handschrift gibt es oft eine Schriftneigung, die zu Überlappungen der Buchstaben führt. Diese Überlappung macht es schwieriger den Buchstaben zu segmentieren, aber dies kann kompensiert werden. In der arabischen Handschrift gibt es viele Überlappung und oft keine regelmäßig Neigung. Damit ist eine Kompensation der Neigung nur schwer möglich. Auch gibt es keine Pausen zwischen den einzelnen Buchstaben. Diese Unterschiede macht die Erkennung von Arabisch viel aufwendiger.

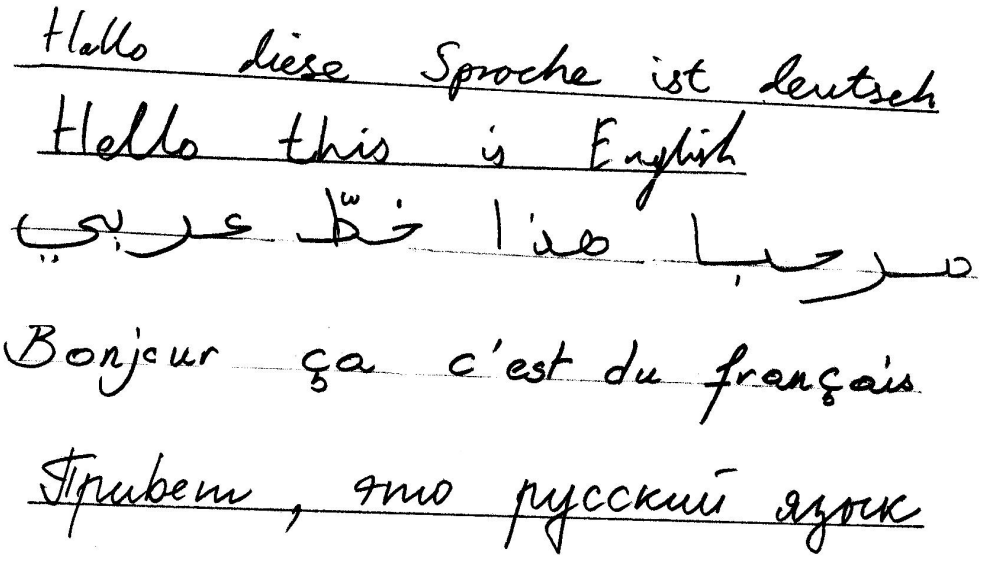


Bild 2 Beispiel von Handschrift.

Die Basislinie ist eine sehr wichtiger Teil des Normierungssprozesses, welcher der Erkennung vorausgeht. Es ist die Linie auf der die Wörter oder Sätze geschrieben sind. Normalerweise wird die Basislinie als ein Hilfsmittel benutzt, um das Word in die Mitte des Bildes zu schieben, bevor eine Erkennung versucht wird. Wenn die Basisline geschätzt ist (Bild 2), fällt auf, dass die meisten der Buchstaben auf der Basisline sitzen. Bei der arabischen Handschrift passiert das im Prinzip auch. Allerdings werden viele Buchstabenteile parallel zur Basislinie geschrieben, so dass leicht eine falsche Basislinie ermittelt wird. Es ist somit oft schwierig die richtige Basesline zu finden. Eine weitere Schwierigkeit ist es das jeder Buchstabe vier mögliche Formen haben kann. Die Form hängt von der Stelle im Wort ab (am Ende, Mittel oder Anfang). Wenn all diese Faktoren zusammengefasst betrachtet werden, wird klar, warum Arabisch so schwierig für Software zu erkennen ist und warum es noch nicht so viel Software gibt. Unten gibt es eine Liste der arabischen Buchstaben (Bild 3).

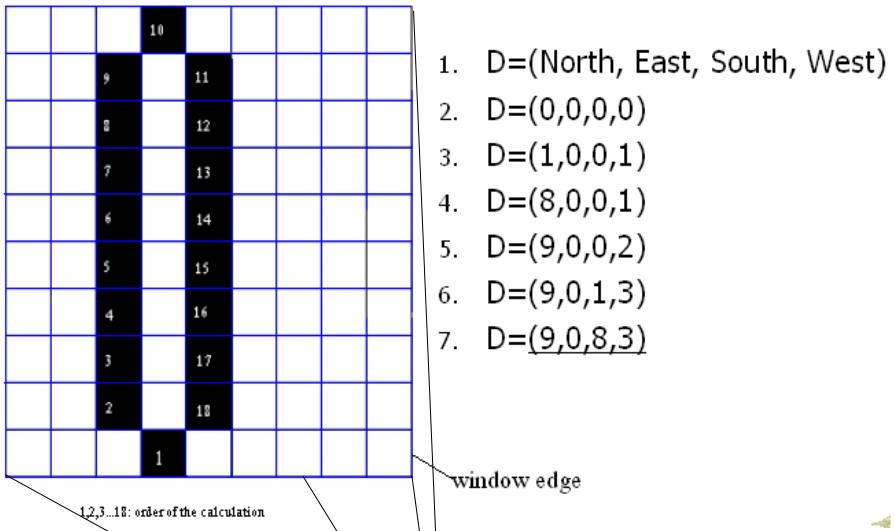
IPA	Value	Name	Final	Medial	Initial	Isolated	IPA	Value	Name	Final	Medial	Initial	Isolated
[ɖ]	ḍ	ḍād	ض	ضـ	ضـ	ض	[ʔ]	ʾ(a)	alif	ا	—	—	ا
[ʈ]	ṭ	ṭāʾ	ط	ط	طـ	ط	[b]	b	bāʾ	ب	بـ	بـ	ب
[ʐ]	ẓ	ẓāʾ	ظ	ظ	ظـ	ظ	[t]	t	tāʾ	ت	تـ	تـ	ت
[ʕ]	ʿ	ʿayn	ع	ـع	ـعـ	ع	[θ]	th	thāʾ	ث	ثـ	ثـ	ث
[ɣ]	gh	ghayn	غ	ـغ	ـغـ	غ	[ʤ]	j	jīm	ج	ـج	جـ	ج
[f]	f	fāʾ	ف	ـف	فـ	ف	[ħ]	ḥ	ḥāʾ	ح	ـح	حـ	ح
[q]	q	qāf	ق	ـق	قـ	ق	[x]	kh	khāʾ	خ	ـخ	خـ	خ
[k]	k	kāf	ك	ـك	كـ	ك	[d]	d	dāl	د	—	—	د
[l]	l	lām	ل	ـل	لـ	ل	[ð]	dh	dhāl	ذ	—	—	ذ
[m]	m	mīm	م	ـم	مـ	م	[r]	r	rāʾ	ر	—	—	ر
[n]	n	nūn	ن	ـن	نـ	ن	[z]	z	zāy	ز	—	—	ز
[h]	h	hāʾ	ه	ـه	هـ	ه	[s]	s	sīn	س	ـس	سـ	س
[w]	w	wāw	و	—	—	و	[ʃ]	š	shīn	ش	ـش	شـ	ش
[j]	y	yāʾ	ي	ـي	يـ	ي	[s]	ṣ	ṣād	ص	ـص	صـ	ص

Das arabische Alphabet von [10] Bild 3

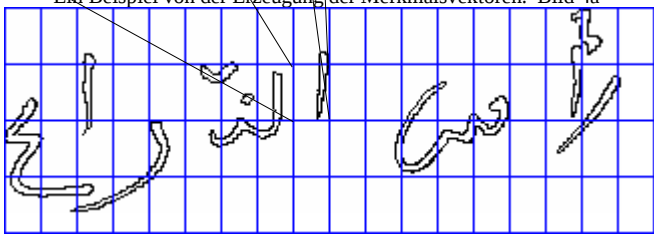
1.3 Zusammenfassen der Referenzarbeit

Dieses Projekt basiert auf der Arbeit von M. Dehgahn, K Faez, M. Ahmadi, M. Shridhar vom Februar 2000. In dieser Veröffentlichung wurde ein arabisches Handschrifterkennungssystem vorgestellt. Dabei kam eine spezielle Merkmalsextraktionsmethode zum Einsatz. Diese Methode extrahiert die Merkmale mit einer Fenstermethode. Das Bild wird in Fenstern aufgeteilt, wobei der Schriftzug bereits im Kettencode vorliegt. Der Kettencode wird innerhalb von jedem Fenster extrahiert und ein Merkmalvektor wird zusammengestellt. Dieser Vektor hat vier Komponente (oder Richtungen) Nord, Ost,West und Süd. Von jedem Konturpunkt wird die Richtung zu dem darauf folgenden Konturpunkt bestimmt (Nord, Ost, West oder Süd). Jedesmal wenn eine der Richtungen ermittelt wurde, wird zu dem jeweiligen Vektor Eins dazu addiert. In Bild 4 ist ein Beispiel dieser Methode abgebildet. Weiterhin enthält Bild 5 ein Beispiel dieser Fenster Methode, wobei fünf übereinander liegende Fenster veranschaulicht sind. Die Fensterweite wurde in der Veröffentlichung von Dehgahn auf die doppelte durchschnittliche Linienbreite festgelegt. Die Fenster überlappen sich mit 50%. Es gibt jedoch keine vertikale Überlappungen. Die Autoren von [1] schrieben, dass über 400000 Merkmalvektoren erzeugt wurden. Die Autoren haben ein SOFM

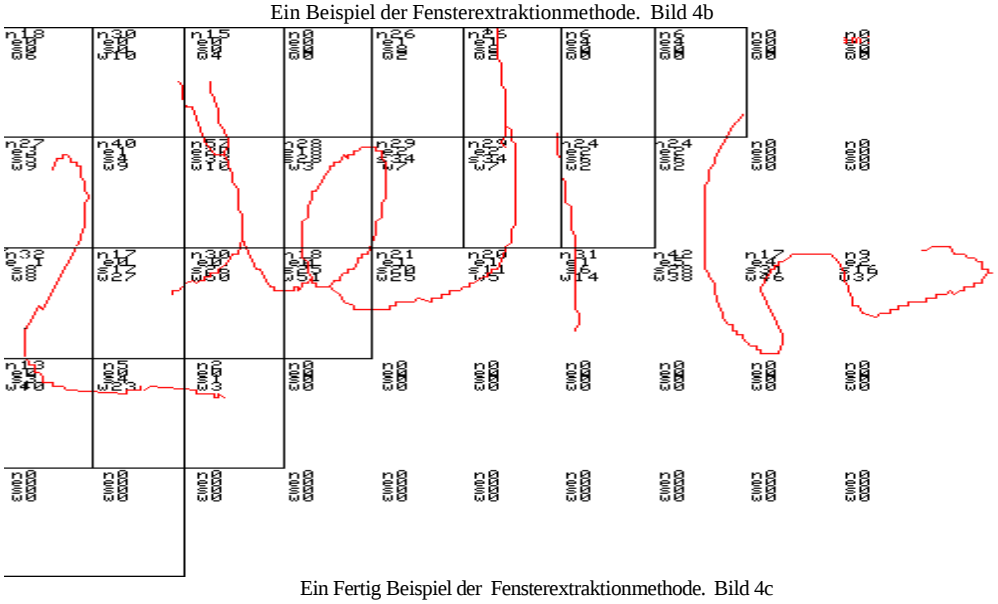
(Self Organizing Featrure Map) [1] entwickelt, um die Zahl der Merkmalvektoren auf die wichtigsten zu beschränken. Danach wurde ein HMM (Hidden Markov Model) angewendet. Die Autoren haben herausgefunden, dass fünf senkrechte Fenster die besten Ergebnis liefern. Die Erkennungsrate konnte mit fünf Fenstern, im vergleich zu vier Fenstern, nur minimal gesteigert werden, der Rechenaufwand stieg jedoch enorm an. Aus diesem Grund wurde sich für die Verwendung von vier Fenstern entschieden.



Ein Beispiel von der Erzeugung der Merkmalsvektoren. Bild 4a



Ein Beispiel der Fensterextraktionmethode. Bild 4b



Ein Fertig Beispiel der Fensterextraktionmethode. Bild 4c

1.4 Vorgeschlagene Arbeit

Hauptgegenstand der Untersuchung ist die Merkmalsextraktionsmethode, die auf der Fenster Methode gründet und mit der Merkmale, wie sie in [1] verwendet, extrahiert werden. Damit soll untersucht werden, in wieweit die Abhängigkeit von der Baseline unterdrückt werden kann, und wie viel Einfluss die Genauigkeit der Basisline auf die Erkennungsrate hat.
Zur Untersuchung der Methode werden Teile der am IFN (Institut für Nachrichtentechnik) entwickelten Erkenner-Software benutzt, die auf Semi Continuos Hidden Markov Model beruht.

Die folgenden Abhängigkeiten werden in dieser Arbeit untersucht:
- Zahl der vertikalen Fenster
- horizontale und vertikale Überlappung der Fenster
- Die Auswirkung der Linienbreite – (Fensterbeite = 2 x Linienbreite)

Nur die Anzahl der vertikalen Fenster wurde in der Veröffentlichung untersucht.

1.5 Wort Vorverarbeitung, Normierung, Entfernung von Störungen

Die Autoren von [1] haben erstens ein morphologisches „closing“ und danach „opening“ mit einem 3x3 strukturierenden Element durchgeführt um Störungen (Rauschen) im Bild zu reduzieren. Dann wurde eine Basislinienkorrektur durchgeführt, um ein schiefes Bild geradezurücken. Die dazu nötige Basislinie wurde aus einem horizontales Projektionshistogramm geschätzt, das bei kleinen Winkeländerungen von der Normalen berechnet wurde. Danach wurde das Bild um diesen Winkel korrigiert und leere Reihen wurden hinzugefügt um das Word in die Mitte des Bildes zu schieben. Danach wurde das Bild zu Chaincode konvertiert.

Das entfernen von Störungen war in diesem Projekt nicht nötig, da die Bilder in der benutzten arabischen Datensammlung weitgehend störungsfrei sind. Die Schriftzüge lagen auch als Skelettdaten vor. Deshalb werden die Schritte, die im oberen Abschnitt beschrieben wurden, nicht durchgeführt. Das Ziel dieses Projekts ist, den Algorithmus so umzustellen das eine nicht hundertprozentig korrekte Basislinie, nicht mehr so einen großen negativen Einfluss auf die Erkennungsrate hat. Es werden dazu zwei Datensätze mit Wörter benötigt. Der erste Datensatz enthält die Wörter, die auf einer genauen Basislinie ausgerichtet waren. Der zweite Datensatz enthält Wörtern die auf einer schlecht definierten Basislinie beruhen. Die gesamte IFN/ENIT Datensammlung beinhaltet so genannte „ground truth“ Daten. Diese enthalten wiederum eine von Hand normierte Baseline (immer richtige). Diese Basisline wird benutzt um den Datensatz der gut normierten Wörter zu erzeugen. Zum Erzeugen des schlecht normierten Datensatz werden Basisliniendaten genutzt, welche zuvor ein Algorithmus zur Basislinienschätzung berechnet hat [9]. So gibt es insgesamt zwei Datensätze mit Wörtern, eine mit gut normierten Wörtern (perfekt von Hand erzeugt) und eine mit schlecht normierten Wörtern (durch einen Rechner erzeugt, möglicherweise fehlerhaft).