ELECTRONIC PUBLISHING, VOL. 6(3), 273–282 (SEPTEMBER 1993)

Improving the recognition accuracy of text recognition systems using typographical constraints

RENÉ SENNHAUSER

MultiMedia Laboratory
Department of Computer Science
University of Zürich
Winterthurerstrasse 190
CH–8057 Zürich, Switzerland

email: sennhaus@ifi.unizh.ch

SUMMARY

Spelling correction techniques can be used to improve the recognition accuracy of text recognition systems. In this paper a new spelling-error model is proposed that is especially suited to the correction of recognition errors occurring during the recognition of printed documents. An implementation of this model is described that exploits typographical constraints derived from character shapes. In particular, the fact is used that vertical strokes in character images are seldom misrecognised. Experimental results show: 1) that the sizes of candidate word sets are substantially reduced; and 2) that the probability that the wrong candidate word is chosen is reduced by an average factor of approximately 2 when compared to spelling correction techniques without the use of typographical constraints.

KEY WORDS Text recognition Recognition accuracy Spelling correction Typographical constraints Stem matching Typographical distance measure

1 INTRODUCTION

Text recognition systems are of immediate necessity for efficient input of the huge amounts of printed documents processed in modern business transactions. Manual input of printed documents into a computer is very time-consuming and error-prone, and therefore cost-intensive. To reduce costs for document input, reading machines have been conceived and developed for many years. Nevertheless, the spread of these machines has been slower than expected. This is mainly due to deficiencies in recognition accuracy. Current text recognition systems achieve 100% recognition accuracy only for shape-optimised fonts of numerals as used in OCR-A and OCR-B applications, but show a markedly decreased recognition accuracy for general documents [1].

Spelling correction of the recognition result can improve recognition accuracy. In this approach, every recognised word is verified using an electronic dictionary, and if the word is not among the dictionary words an attempt is made to correct it. Although the most advanced text recognition systems include spelling correction components, there is much room for improvement.

CCC 0894–3982/93/030273–10
Received 1 September 1993
Revised 3 December 1993

Table 1. Number of individual words with rejection or substitution errors

Font	*Error type*	7 pt	8 pt	9 pt	10 pt	11 pt	12 pt	13 pt	14 pt
Courier	Rejection	123	9	4	0	5	1	5	15
	Substitution	4540	74	34	79	642	141	332	74
	Total	4663	83	38	79	647	142	337	89
Helvetica	Rejection	37	62	0	11	4	14	26	41
	Substitution	551	1480	81	70	146	83	138	229
	Total	588	1542	81	81	150	97	164	270
Times	Rejection	273	14	17	0	25	20	31	50
	Substitution	2611	1775	251	301	661	47	73	319
	Total	2884	1789	268	301	686	67	104	369

This paper is organised as follows. Section 2 gives an overview of text recognition systems. In Section 3, a new spelling-error model is introduced that is especially suited to the correction of recognition errors. In Section 4, an implementation of the new spelling-error model is presented. The use of typographical constraints to improve the correction accuracy is proposed in Section 5. Finally, in Section 6 experiments with the new techniques are described.

2 TEXT RECOGNITION SYSTEMS

The main task of text recognition systems is to convert images of printed characters into identifying codes (e.g. ASCII or EBCDIC). This conversion is conceptually done in six steps: 1) digitisation, 2) image pre-processing, 3) line segmentation, 4) character segmentation, 5) character classification, and 6) postprocessing.

Recognition errors can either be classified by the processing steps where the error occurs, or by the effect the error has on the recognised text. During the conversion of a document image into its textual representation, recognition errors occur mainly in the character segmentation (3, 4) and classification (5) steps. Characters are wrongly segmented (*segmentation error*) and wrongly classified or not at all (*classification error*). One or several characters of a word are replaced by other characters (*substitution error*) or marked as unclassifiable (*rejection error*).

Substitution and rejection errors are not independent of each other. If the number of substitution errors is minimised in a text recognition system the number of rejection errors increases, and vice versa. In general, the number of substitution errors is much larger than the number of rejection errors. This fact was confirmed with an experiment in which a document with 12885 words was printed in three different fonts and eight font sizes, then scanned using a 300-dpi page scanner, and finally recognised by a state-of-the-art commercial text recognition system. The recognition process was tuned to minimise the number of rejection errors. The resulting number of substitution and rejection errors is shown separately for each font and font size in Table 1.

From Table 1, it can be seen that the number of substitution errors is much larger than the number of rejection errors. On average, there are 11.5 times more substitution errors than rejection errors in the above experiments.

The total number of recognition errors is about the same size for a broad spectrum of font sizes in the three fonts examined. However, if the font sizes become smaller than 8 or 9 pt, the number of recognition errors grows rapidly.

3 SPELLING CORRECTION

Recognition errors in individual words can often be corrected because they contain redundancy. For rejection errors the error position is clearly determined. A substitution error can only be found if a correct word is garbled into an unorthographic one. The error position can then be determined by spell-checking of isolated words. An individual word is assumed to be correctly recognised if it is listed in a dictionary. Otherwise, it is potentially misrecognised. In this case, words with the greatest possible similarity according to some measure can be selected from the dictionary as candidate words (spelling correction). If there is only a single word with the greatest possible similarity, the garbled word can be corrected automatically.

Every spelling correction method comprises four components:

Spelling-error model. The spelling-error model describes how words can be garbled by a specific input method.

Dictionary. This component defines all the (orthographically) valid words which are accepted by the spelling corrector. Preferably, a dictionary should contain as many words as possible, so that only a few orthographically correct words are marked as invalid. Unfortunately, the probability that a garbled word is contained in the dictionary increases with increasing dictionary size [2].

Candidate word generation. To correct garbled words, those dictionary words have to be selected that possibly could be the correct word (candidate words).

Candidate word ranking. The candidate words have to be ordered so that the most likely correct candidate word is ranked highest. This requires a procedure to compute a similarity measure between a garbled word and every candidate word.

Most published spelling correction systems have adopted the single-error model of Damerau [3] where only one of the following four types of spelling errors can occur: single insertion, single deletion, single substitution, or transposition of two characters. This model is reported to cover over 80% of all misspellings. However, Damerau's model does not take into account character segmentation errors occurring in character recognition systems. Therefore, a new spelling-error model is proposed in this paper that is a superset of Damerau's model. There are four error categories in this new model:

C *error* (Case error). A word has been correctly recognised except for the case of some of its characters, e.g. 'harmOnIca' instead of 'harmonica'.

S *error* (Single error). Up to three adjacent characters have been garbled at one error position in a word, e.g. 'harrnonica' instead of 'harmonica'. This category has been derived empirically, since more than 90% of all garbled words have been found to contain just one error which extends over at most three characters.

M *error* (Multiple error). Several **S** errors have occurred in a word, e.g. 'hatmonka' instead of 'harmonica'.

R *error* (Real-word error). A word has been garbled into another valid word, e.g. 'harmonics' instead of 'harmonica'.

Table 2. Classes of **S** errors

Error order	*Error description*	*Error class*	*Garbled word*	*Correct word*
1	1 character inserted	1INS	Abladjung	Abladung
	1 character deleted	1DEL	ravel	travel
	1 character substituted	1SUB	runlime	runtime
	1 character rejected	1REJ	mea˜ured	measured
2	2 characters substituted	2SUB	folmd	found
	2 characters rejected	2REJ	Auftrags˜lm	Auftragsfilm
	1 character broken into 2	1BRK2	approacll	approach
	2 characters melted into 1	2MLT1	mles	rules
3	3 characters substituted	3SUB	–	–
	3 characters rejected	3REJ	alkoholabhän˜er	alkoholabhängiger
	1 character broken into 3	1BRK3	ABS'I'RACT	ABSTRACT
	3 characters melted into 1	3MLT1	omine	offline
	2 characters broken into 3	2BRK3	Grihlgasse	Grüngasse
	3 characters melted into 2	3MLT2	nmtime	runtime

S errors are further divided into 14 error classes, which can be grouped according to the number of characters involved (Table 2). The number of characters involved gives the error order. Errors of order 1 correspond to Damerau's single errors, except for transposition errors which belong to error order 2 in the new model (2SUB). To illustrate the error classes, examples are given that have been found in recognition experiments.

The frequency of occurrence of **C**, **S**, **M**, and **R** errors shown in Table 3 has been determined in the same experiment that was carried out to compute the number of substitution and rejection errors. **C** and **S** errors cover more than 90% of all recognition errors. This paper therefore concentrates on the correction of **C** and **S** errors.

4 TECHNICAL ISSUES

The extended spelling-error model has been implemented using an extended version of the spelling corrector proposed by Takahashi et al. [4]. This spelling corrector is designed to handle single errors according to Damerau's spelling-error model, and therefore had to be extended to cover **S** errors. To generate candidate words efficiently, attributes are computed in this spelling corrector for each dictionary word and for each garbled word. The candidate word set then encompasses all dictionary words which have at least one attribute and the first and/or last character in common with the garbled word. Takahashi et al. proposed computation of the attributes as follows (curly braces { } denote an unordered set, whereas brackets [ ] indicate an ordered set):

- For each word, compute a *character set* CS containing all the different characters in the word. Example: CS = {`e,x,a,m,p,l`} for the word `example`.
- Rank the characters in the character set CS according to the infrequency of occurrence of each character. This yields a *ranked character set* RCS. The infrequency

Table 3. Frequency of occurrence of **C**, **S**, **M**, and **R** errors

Font	*Error class*	7 pt	8 pt	9 pt	10 pt	11 pt	12 pt	13 pt	14 pt
Courier	**C**	0	0	1	1	0	0	0	0
	S	4302	83	37	75	611	141	295	88
	M	356	0	0	3	36	1	42	1
	R	5	0	0	0	0	0	0	0
	Total errors	4663	83	38	79	647	142	337	89
Helvetica	**C**	36	42	0	3	10	14	14	40
	S	505	1395	76	73	127	83	139	202
	M	37	74	5	5	13	0	11	28
	R	10	31	0	0	0	0	0	0
	Total errors	588	1542	81	81	150	97	164	270
Times	**C**	39	18	13	10	2	30	30	12
	S	2574	1639	245	288	661	31	51	289
	M	238	93	10	1	8	6	22	68
	R	33	39	0	2	15	0	1	0
	Total errors	2884	1789	268	301	686	67	104	369

of occurrence is derived from the dictionary used, without consideration of word frequencies. Example: RCS = [x,p,m,l,a,e] for the word example.

- From the ranked character set RCS select all characters ranked *N*th or earlier. If there are less than *N* characters, fill the remainder with blanks. This yields the *key character set* KCS of a word. Example: the KCS for the word example is [x,p,m,l] if $N = 4$.
- Each combination of $M < N$ characters in the KCS defines an attribute. Hence there are $\binom{N}{M}$ attributes per word. Example: the word example has the following four attributes {[x,p,m],[x,p,l],[x,m,l],[p,m,l]} if $M = 3$ and $N = 4$.

The candidate words are then ordered according to their similarities to the garbled word. Takahashi *et al*. proposed a simplified variant of a minimum edit distance function [5] as a similarity measure, and called it *simple distance*. This function defines the similarity between two words as the minimum total cost if the two words are compared character by character. The more similar two words are, the lower their simple distance score is. Two extension have been made to the original spelling corrector of Takahashi *et al*.:

- The parameters *N* and *M* had to be chosen such that the candidate word set contains the correct word if it is garbled by an **S** error.
- The simple distance had to be extended. Instead of a single character comparison, sequences of up to three characters have to be compared.

To generate candidate words for a word with *e* adjacent characters garbled, the difference $N - M$ must be greater than or equal to *e* [4]. Within this constraint, *N* and *M* should be chosen such that the number of attributes per word is small and the total number of attributes is large. The first criterion ensures that the generated candidate word sets are small, because only a small number of attributes are consulted. The second criterion guarantees that the number of words per attribute is small, because all words can be distributed among more classes (attributes).

Table 4. The dependence of ρ on the parameters N and M

	1 character involved			*2 characters involved*			*3 characters involved*		
N	4	5	6	4	5	6	5	6	7
M	3	4	5	2	3	4	2	3	4
Attributes/word	4	5	6	6	10	15	10	20	35
Total attributes	2951	17901	83681	351	2951	17901	351	2951	17901
$\rho \times 10^{-3}$	1.36	0.28	0.07	17.09	3.33	0.84	28.49	6.78	1.96

To determine appropriate values for N and M, a fraction ρ has been defined as

$$\rho = \frac{\text{attributes per word}}{\text{total attributes}}, \quad \text{where}$$

$$\text{attributes per word} = \binom{N}{M}, \quad \text{and}$$

$$\text{total attributes} = \binom{27}{M} + \sum_{i=1}^{M-2} \binom{26}{i}$$

For the above formula the alphabet comprises the 26 lowercase letters and the space character. If the number of attributes per word is small and the total number of attributes is large, ρ also becomes small. Therefore, parameters N and M should be chosen such that ρ becomes small (Table 4).

Although values of 7 and 4 for N and M respectively would be perfect for correcting **S** errors, 35 attributes per word is rather too many. As a compromise the next best values for N and M have been chosen: $N = 6$ and $M = 3$. Experiments with this spelling corrector revealed that:

- The number of candidate words grows with increasing error order.
- The number of equally-ranked candidate words increases.

The large candidate word set and the coarse ranking procedure mean that for many garbled words too many and too vaguely ordered spelling suggestions are generated if no additional restricting criteria are used.

5 TYPOGRAPHICAL CONSTRAINTS

It has been observed that vertical strokes are seldom misrecognised during character classification. This fact can be exploited to restrict the candidate word set and rank it more accurately. In the following, this fact is referred to as *typographical constraint*, and can be used as the basis for a similarity measure between words. Two typographical distance measures are proposed to re-rank the candidate word set, namely *StemMatch* and *Feature-Distance*. The best and second-best re-ranked candidate words of error order 1, 2, and 3 are then selected as the final candidate word set.

Table 5. Number of feature stems *fs* per character for the ASCII alphabet

char	*fs*	char	*fs*	char	*fs*	char	*fs*	char	*fs*	char	*fs*
space	–	0	2	@	0	P	1	‘	0	p	2
!	1	1	1	A	0	Q	2	a	2	q	2
"	0	2	0	B	2	R	2	b	2	r	1
#	0	3	1	C	1	S	0	c	1	s	0
$	1	4	1	D	2	T	1	d	2	t	1
%	0	5	0	E	1	U	2	e	2	u	2
&	0	6	2	F	1	V	0	f	1	v	0
’	0	7	0	G	2	W	0	g	2	w	0
(	1	8	2	H	2	X	0	h	2	x	0
)	1	9	2	I	1	Y	0	i	1	y	0
*	1	:	1	J	1	Z	0	j	1	z	0
+	1	;	1	K	1	[	1	k	1	{	1
,	0	<	0	L	1	\	1	l	1	|	1
-	0	=	0	M	2	]	1	m	3	}	1
.	0	>	0	N	2	^	0	n	2	~	0
/	1	?	0	O	2	_	0	o	2	DEL	–

5.1 Stem matching *(StemMatch)*

For substitution errors, the generated candidate word set can be restricted by using the fact that vertical strokes and nearly-vertical strokes are seldom misrecognised. For easier reference, such strokes are referred to here as *feature stems*.

Since feature stems of a character are represented as peaks in the projection of the character’s pixels on the *x*-axis, the number of feature stems has been determined as follows (Table 5):

- Every character of the ASCII alphabet has been rendered as a bitmap.
- For every character image, a histogram has been computed which counts the number of pixels at every *x*-position.
- A single threshold value has been determined for all the histograms.
- The number of transitions in a character’s histogram from below to above the threshold then defines the number of feature stems for the character.

There is one exception: although the histogram of the characters / and \ does not exceed the threshold the number of feature stems for these characters is set to 1, since the feature stems of these characters are seldom misrecognised.

A candidate word and a garbled word should have an equal number of feature stems. Since the proposed technique is designed to correct **S** errors, sequences of mismatching characters are uniquely characterised by the first mismatching character from the left and the first mismatching character from the right. The more the number of feature stems for the mismatching parts of the garbled word agrees with those of a candidate word, the more likely the candidate word is to be the correct word.

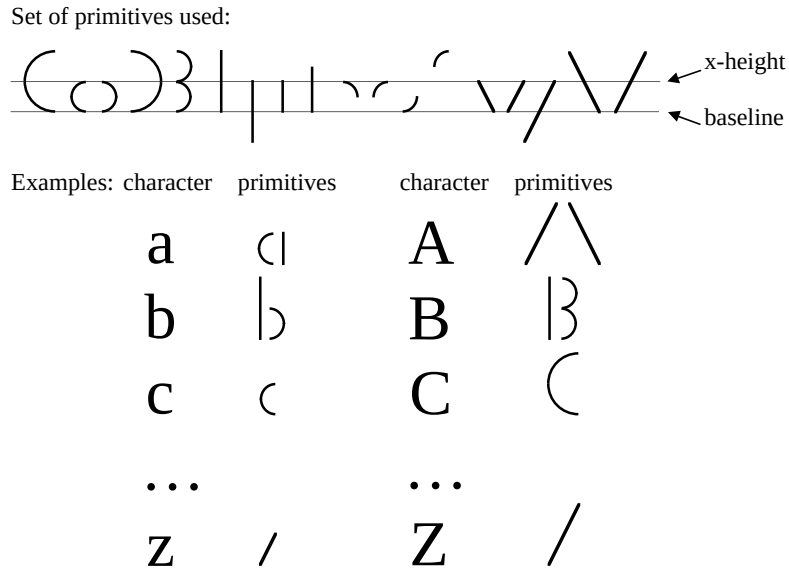

Figure 1. Primitives for modelling characters

Given two words

$$\begin{aligned} \mathtt{S} &= \langle s_0 = \sqcup, s_1, s_2, \ldots, s_m, s_{m+1} = \sqcup \rangle \quad \text{and} \\ \mathtt{T} &= \langle t_0 = \sqcup, t_1, t_2, \ldots, t_n, t_{n+1} = \sqcup \rangle \end{aligned}$$

where $\sqcup$ denotes a space,

$$\begin{aligned} s_i &= t_i \quad \text{for } 0 \le i \le l, \quad \text{and} \\ s_{m+1-j} &= t_{n+1-j} \quad \text{for } 0 \le j \le r \end{aligned}$$

i.e. the first l and the last r characters are equal, and

$$\begin{aligned} \mathtt{S}_{mis} &= s_{l+1}, s_{l+2}, \ldots, s_{m+1-r} \quad \text{and} \\ \mathtt{T}_{mis} &= t_{l+1}, t_{l+2}, \ldots, t_{n+1-r} \end{aligned}$$

are mismatched. Then *StemMatch* is defined as

$$\mathit{StemMatch}(\mathtt{S},\mathtt{T}) = \mathrm{abs}(\mathit{CountStems}[\mathtt{S}_{mis}] - \mathit{CountStems}[\mathtt{T}_{mis}])$$

Example: if the spelling corrector generates the candidate words `comes` and `corner` for the garbled word `comer`, `corner` is more likely to be the correct word, because the number of feature stems in `s` and `r` differs by 1 while the number of feature stems in `rn` and `m` differs by 0.

5.2 The typographic distance measure *(FeatureDistance)*

The stem matching technique described above has been refined to a typographic distance measure *FeatureDistance* where the single stem feature was generalised to 18 *typographic*

Table 6. Example for *FeatureDistance*

	garbled word	*candidate word*
	golclcn	golden
mismatched characters	clc	de
sequence of primitives	c | c	c | c

primitives. In this method characters are modelled by a sequence of primitives (Figure 1). To compare a garbled word against a candidate word, the mismatching characters in both words are represented by their sequences of primitives. A cost function defines the costs of transforming any primitive into another primitive, deleting a primitive or inserting a primitive. The distance between a garbled word and a candidate word is then defined as the cost of transforming the sequence of primitives in the garbled word into the sequence of primitives in the candidate word.

In the following, the same definitions for $\mathtt{S}_{mis}$ and $\mathtt{T}_{mis}$ are used as in subsection 5.1. If $G[s_i,s_{i+1},\ldots,s_j]$ is the concatenation of the sequences of primitives of characters $s_i,s_{i+1},\ldots,s_j$, then the typographic metric *FeatureDistance* between S and T is defined as

$$\mathit{FeatureDistance}(\mathtt{S},\mathtt{T}) = \mathtt{cost}(G[\mathtt{S}_{mis}] \rightarrow G[\mathtt{T}_{mis}])$$

An example is shown in Table 6. The *FeatureDistance* for this example is given by

$$\mathit{FeatureDistance}(\text{golclcn},\text{golden}) = \mathtt{cost}(G[\text{clc}] \rightarrow G[\text{de}]) = \mathtt{cost}(\mathtt{c}|\mathtt{c} \rightarrow \mathtt{c}|\mathtt{c}) = 0$$

6 EXPERIMENTS

Experiments were made to test the effect of typographical constraints on the correction accuracy. All the recognition errors that occurred in the experiment to determine the number of substitution and rejection errors were used as input to the new spelling correction techniques. Since the new techniques are not designed to handle **M** and **R** errors, these error categories were extracted. The reference dictionary consisted of the 2755 different words that occurred in the document used in the experiment.

In Table 7, the first number in each column indicates the correction accuracy, i.e. the probability that the highest-ranked word is the correct word. The second number gives the

Table 7. Correction results for different spelling correction techniques. The correction accuracy is denoted by CA and the reduction rate by RR

Recognition errors	*Tspell1*		*Tspell3*		*StemMatch*		*FeatureDistance*	
from documents	CA	RR	CA	RR	CA	RR	CA	RR
Courier 7–14 pt (1363 words)	91.9%	2.5	91.2%	1.0	93.8%	1.7	95.2%	6.9
Helvetica 7–14 pt (1299 words)	69.4%	2.0	71.1%	1.0	71.1%	1.8	85.1%	6.9
Times 7–14 pt (2634 words)	64.8%	2.0	66.5%	1.0	58.9%	1.8	84.9%	6.8

reduction rate, which is defined as the proportion of the average number of highest-ranked candidate words to the average number of all generated candidate words.

Column *Tspell1* gives the results for the spelling corrector of Takahashi *et al*., which is capable of correcting single errors. Column *Tspell3* lists the results for the extended version, which is designed to correct **C** and **S** errors. Columns *StemMatch* and *FeatureDistance* indicate the results achieved if the corresponding typographical constraints are applied to restrict further and rank more accurately the candidate word set computed by *Tspell3*.

From Table 7, the following can be seen. The correction accuracy is about the same for *Tspell1* and *Tspell3*, although *Tspell3* handles a larger set of recognition errors. If *StemMatch* is used, the correction accuracy increases slightly for the documents printed in Courier and Helvetica, but it decreases for the documents printed in Times. Re-ranking by *FeatureDistance* substantially improves the correction accuracy in the experiments made. The probability that the wrong candidate word is chosen is reduced by a factor of 1.8 for the documents printed in Courier, 2.1 for the documents printed in Helvetica, and 2.3 for the documents printed in Times. At the same time, the reduction rate increases to 6.8 or better. The size of the candidate word sets generated is therefore nearly 7 times smaller if *FeatureDistance* is used.

7 CONCLUSIONS

The accuracy of text recognition systems can be improved by spelling correction techniques. In this paper, typographical constraints from character shapes have been investigated to reduce and rank more accurately the number of candidate words generated by a spelling corrector. Two different typographical distance measures have been proposed: a simple stem matching technique (*StemMatch*), and a refined technique (*FeatureDistance*). The highest correction accuracy has been achieved for *FeatureDistance*, where the probability that the wrong candidate word is chosen is reduced by an average factor of approximately 2 when compared to spelling correction techniques without the use of typographical constraints.

ACKNOWLEDGEMENTS

This work is supported by the *Kommission zur Förderung der wissenschaftlichen Forschung des Eidgenössischen Volkswirtschaftsdepartements der Schweiz* under grant no. 2270.2, and also by the Swiss Life Insurance and Pension Company, Switzerland.

REFERENCES

1. G. Nagy, 'At the frontiers of OCR', *Proceedings of the IEEE*, **80**(7), 1093–1100, (1992).
2. J. L. Peterson, 'A note on undetected typing errors', *Communications of the ACM*, **29**(7), 633–637, (1986).
3. F. J. Damerau, 'A technique for computer detection and correction of spelling errors', *Communications of the ACM*, **7**(3), 171–176, (1964).
4. H. Takahashi, N. Itoh, T. Amano, and A. Yamashita, 'A spelling correction method and its application to an OCR system', *Pattern Recognition*, **23**(3/4), 363–377, (1990).
5. R. A. Wagner and M. J. Fischer, 'The string-to-string correction problem', *Journal of the Association for Computing Machinery*, **21**(1), 168–173, (1974).