Sādhanā Vol. 27, Part 1, February 2002, pp. 99–111. © Printed in India

A post-processor for Gurmukhi OCR

G S LEHAL[1] and CHANDAN SINGH[2]

[1] Department of Computer Science and Engineering, Thapar Institute of Engineering & Technology, Patiala 147 004, India
[2] Department of Computer Science and Engineering, Punjabi University, Patiala 147 002, India
e-mail: gslehal@mailcity.com

Abstract. A post-processing system for OCR of Gurmukhi script has been developed. Statistical information of Punjabi language syllable combinations, corpora look-up and certain heuristics based on Punjabi grammar rules have been combined to design the post-processor. An improvement of 3% in recognition rate, from 94.35% to 97.34%, has been reported on clean images using the post-processing techniques.

Keywords. Post-processing; Gurmukhi script; corpus; OCR.

1. Introduction

The objective of post-processing is to correct errors or resolve ambiguities in OCR results by using contextual information. There are a number of levels at which context may be operative. It can be at the word level, at the sentence level and at the level of semantics. The most common post-processing technique which operates at the word level is the dictionary look-up method (Wells *et al* 1990; Mayes *et al* 1991). The output of the OCR is compared to the system's built-in dictionary (lexicon) and candidates are generated. According to the difference between the output of the OCR and the output of the dictionary look-up, the numbers denoting the confidence level in the correct classification are modified. The output sequence of suitable candidates is then ordered and the best candidate selected. Another very common post processing technique is based on statistical information about the language (Riseman & Hanson 1974; Suen 1979; Hull & Srihari 1982; Yannakondakis *et al* 1990; Tong & Evans 1996). In this method, an n-gram is used to filter out unacceptable letter string candidates from the recognizer. An n-gram is a letter string of size n. Two grams are referred to as bigrams, three grams as trigrams and so on; n-gram tables are created from a corpora and the output letter strings are checked against these tables, both to see whether they occur (if not they can be discarded), and, if they do, to use their probability of occurrence to decide which one to choose. Another similar approach is *word collocation* evaluation implemented at OCR output to perform post-processing (Church *et al* 1990). Word collocation tables express the probability of two words being found in the text in the given order. As an

addition the constraints of impossible combinations can be implemented. Word-level post-processing is not sufficient to detect and correct many recognition errors particularly in case of degraded text. One technique for improving post-processing performance is called context-dependent word correction. Passage-level linguistic contextual constraints, above the word level, are utilized in this process. Hong (1995) has used different high-level knowledge sources and visual contextual information in a text image to enhance text recognition in degraded text images such as multiple generation photocopy or facsimile.

Not much literature is available on post-processing techniques for Indian language script recognition systems. Sinha (1987) has developed a rule-based contextual post processor for Devanagri text recognition. Bansal & Sinha (1999) have developed a partitioned word dictionary for correcting optically read Devanagari character strings. The word dictionary is partitioned in order to reduce the search space besides preventing forced match to incorrect words. Word size and the envelop information of words are taken as the main partitioning features.

In this paper we describe a post-processor for improving the recognition rate of an OCR of Gurmukhi script. The complete details of the Gurmukhi OCR system used in the present study are as given earlier (Lehal & Chandan Singh 1999, 2000). We have used a Punjabi corpus, which serves the dual purpose of providing data for statistical analysis of the Punjabi language and also for checking the spelling of a word. The corpus has been partitioned at two levels. At the first level, the corpus is split into seven disjoint subsets based on word length. At the second level, we have used the shape of the word to further segment the subset into a list of visually similar words. We have used a set of robust, font- and character-size independent features for identification of visually similar words. These features are available more or less as by-products of the on-going recognition process and do not necessitate any additional computation. Punjabi grammar rules are also incorporated to check for illegal character combinations such as presence of two consecutive vowels or a word starting with a forbidden consonant or vowel.

2. Characteristics of Gurmukhi script

Gurmukhi script is used primarily for the Punjabi language which is the world's 14th most widely spoken language. Some of the properties of the Gurmukhi script are as below.

Figure 1. Character set of Gurmukhi script.

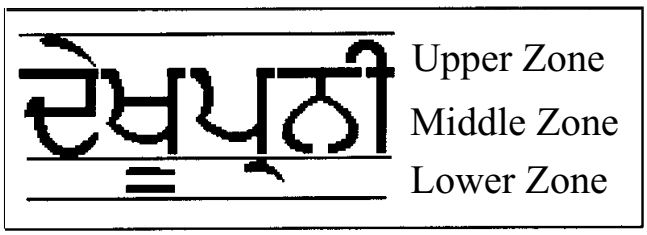


Figure 2. Three zones of a word in Gurmukhi script.

- The Gurmukhi script is cursive and the alphabet consists of 41 consonants, 12 vowels and 3 half characters (figure 1).
- Most of the characters have a horizontal line at the upper part. The letters of a word are connected mostly by this line, called the head line, and thus there is no vertical inter-character gap in the letters of a word. Formation of merged characters is therefore a norm rather than an aberration in the Gurmukhi script
- A word in Gurmukhi script can be partitioned into three horizontal zones (figure 2). The upper zone denotes the region above the head line, where vowels reside, while the middle zone represents the area below the head line where the consonants and some sub-parts of vowels are present. The middle zone is the busiest zone. The lower zone represents the area below the middle zone where some vowels and certain half characters lie in the foot of consonants.
- Bounding boxes of 2 or more characters in a word may intersect or overlap vertically.
- Characters in the lower zone may touch characters in the middle zone.
- There are many topologically similar character pairs in the Gurmukhi script. They can be categorized as below.

(i) Character pairs which after thinning or in noisy conditions appear very similar (ਟ and ਦ, ਤ and ੜ, ਬ and ਥ, ਤ and ੜ, ੍ and ੱ).

(ii) Similar looking character pairs which are only differentiated by whether they are open/closed along the headline (ਸ and ਮ, ਧ and ਪ,ਬ and ਖ).

(iii) Character pairs which are exactly similar in shape and distinguished only by the presence/absence of a dot at the foot of a character (ਸ and ਸ਼, ਖ and ਖ਼, ਜ and ਜ਼, ਫ and ਫ਼, ਗ and ਗ਼).

3. Proposed scheme

We have used a Punjabi corpus for generating the word frequency list, which is the backbone of the post-processing module. The Punjabi corpus used in the present study was provided by the Central Institute of Indian Languages, Mysore. The corpus was developed in the 1990s under funding from the Technical Directorate of Indian Languages, Department of Electronics (DOE), Government of India.
Some of the characteristics of the corpora are as below.

Number of words	**8,43,590**
Number of Characters	**32,72,268**
Number of Unique words	**55,071**

The main steps in the post processing phase are the following.

(1) Create the word frequency list from the Punjabi corpus. The list stores the frequency of occurrence of all words present in the corpus.
(2) Partition the word frequency list into smaller sub-lists based on word size. We have created 7 sub-lists corresponding to word sizes two, three, four, five, six, seven, greater than seven.
(3) Generate from each of the sub-lists a dynamic list of structures, which is based on visually similar characters. This list of structures records the percentage frequency of occurrence of the character in all the positions of a word. This list is combined with the confidence rate of recognition of the recognizer to correct the errors of the recognizer.
(4) Correct the upper dot related errors.
(5) Use Punjabi grammar rules to eliminate illegal character combinations.

Steps 3–5 are explained in detail in the following sections.

4. Creation of visually similar word structure list

As already discussed in the previous section, the corpus is divided into seven sub-sets based on word size. Further in each of these sub-sets, a list of visually similar words is generated. We say that two words are visually similar, if each character in the corresponding position of the two words is visually similar. To decide the visual similarity of two characters, the zonal position of the character and a set of robust features are used. For this purpose, the Gurmukhi character set is divided into 16 sub-sets. Out of the 16 sub-sets, the first ten sub-sets contain the characters present in the middle zone. The middle zone characters are assigned the sub-set by the recognizer using the following font- and size-invariant features. These features have Boolean values.

(*i*) *Number of junctions with the headline:* It is observed that each character present in the middle zone merges with the headline at one or more than one point. For example, the character ਫ has one junction while the character ਪ has two junctions with the headline. This feature is true if the number of junctions is one, else it is false.
(*ii*) *Presence of sidebar:* This feature is true if a vertical line is present on the rightmost side of the sub-symbol, else it is false. For example, this feature is true for the character ਮ while it is false for the character ਕ.
(*iii*) *Presence of a loop excluding the headline:* This feature is true if there is a loop in character image. The loop should not be formed along the headline. This feature is true for the character ਰ but is false for the character ਯ, since the loop of ਯ formed along the headline.
(*iv*) *Loop along the headline:* This feature is true if the character forms a loop with the headline. Examples of characters with this feature are sub-symbols of ਲ and ਯ.

All the members of a sub-set share the same Boolean values of the above mentioned features. For example, for all the members of sub-set 2 (table 1), the value of the first feature is true, second feature is false, third feature is true and fourth feature is false, since all the characters in this sub-set have one branch from the headline, do not have a side bar, contain a loop but not along the headline. The eleventh and twelfth character sub-sets correspond to the upper zone and lower zone characters respectively. We have created separate sub-sets for some of the most frequently occurring characters, which have a very high recognition rate and are not

Table 1. Partitioning of Gurmukhi character set into 16 sub-sets.

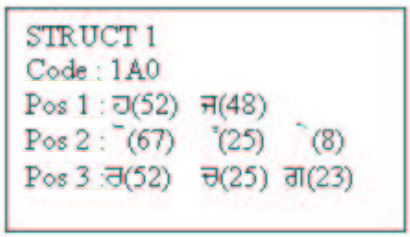
Sub-set no.	Character set
0	ਚ ਰ
1	ਹ ਜ ਜ਼
2	ਕ ਖ਼ ਛ ਠ ੜ ਢ ਫ ਭ ਫ਼
3	ਟ ਣ ਤ ਦ ਨ ਵ ੜ
4	ਖ ਖ਼ ਗ ਗ਼
5	ਥ ਬ
6	ਅ ਘ ਪ ਮ
7	ਸ ਧ ਯ ਸ਼
8	ੳ ੴ
9	ੲ ਝ ਵ਼ ਲ ਲ਼
10	◌ਂ ◌ੰ ◌ੱ ◌ੵ
11	◌ਾ ◌ੌ ◌ੁ ◌ੂ
12	◌ਃ
13	◌ਁ
14	◌ਿ
15	◌ੀ

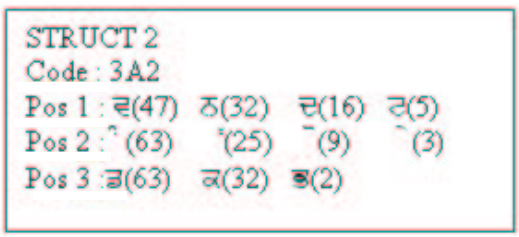
confused with any other character. The thirteenth sub-set contains only the character ◌ਾ. From a statistical analysis of the corpus it was found that the character ◌ਾ is the most frequently occurring character with a frequency of occurrence of 10% and it is very easily recognizable. Similarly the character ◌ਂ , which is just a dot present in the upper zone and hereby referred as *bindi*, is very easily recognizable and not confused with any other character and has 5% frequency of occurrence, is assigned the fourteenth sub-set. The fifteenth and sixteenth character sub-sets consist of ◌ਿ and ◌ੀ characters. These characters, which are present in both upper and lower zones, have a high frequency of occurrence, and are not confused with any other character. The complete sub-sets are shown in table 1.

Each structure represents the visually similar words and the percentage frequency of occurrence of characters in different positions of the word. We call this structure SSS (Shape-based Statistical Structure).These structures are assigned a unique code generated from the sub-list number of the characters and the structures are arranged in sorted order of the code. To clarify how the SSS structure is spawned, we consider the following ten-word frequency list of words of length 3.

Word	Frequency
ਜੋਗ	1400
ਹੇਰ	500
ਹੋਰ	2600
ਜੱਚ	1500
ਨੱਕ	2500
ਦੜ	1600
ਵਡ	4700
ਨੋਕ	700
ਟੋਭ	200
ਟੇਕ	300

From the above frequency list the following two structures are generated:





The first structure is generated using the first four words of the frequency list. These words have visually similar characters in all the three positions. The first character in each of these words belongs to sub-set 1, second character belongs to sub-set 10 and third character belongs to sub-set zero. Thus the code of this structure in hexadecimal format is 1A0, representing the subset of the three character positions. The character ਹ is present in the first position in the first and second word, giving a total frequency 3100 and similarly the character ਸ is present in first position in the third and fourth words, giving a total frequency 2900. The percentage frequencies of occurrence of ਹ and ਸ are thus 52 and 48 respectively. Similarly the percentage frequencies of occurrence of characters in second and third positions is calculated. These structures can then be used to decide between confusing characters. If, for example, the recognizer determines that the first character of a three-lettered word has no sidebar, one branch from the headline and no loops, the second character is present above the headline and the third character has no sidebar, one branch with the headline and one loop, then the second structure could be used to assist the recognition process. It predicts that the most probable characters occurring in the three positions are ਵੰ, and ੜ. This prediction is combined with the results of the recognizer to decide the actual characters.

This recognition scheme is similar to bi-gram and tri-gram post processing analysis but it has the following advantage. In case of bi-gram analysis the prediction of the next character depends upon correct recognition of the previous character. If, for example, the first character is incorrectly recognized then the bi-gram analysis predicts the second character based on the wrongly recognized first character and this result carries on for the subsequent characters. In our current scheme, we have used very robust font- and size-independent features to categorize a character to one of the sub-sets. It was observed that on clean images, the characters were correctly categorized to their sub-sets in 99.82% cases. So even if the first character is not correctly recognized, if its subset is correctly identified, the performance of the post-processor is not affected. The only limitation is that, if the sub-set is not properly distinguished, then the post-processor may not yield correct results.

5. Application of SSS in post processing

The recognizer generates a character distance pair set (CD) $\{(c1, d1), (c2, d2)\}$, where $c1$ represents the best matching character and $d1$ represents the distance of the input character image from the character prototype stored in the training data. Similarly $c2$ and $d2$ represent the second nearest matching character and the distance of the input character image with the character prototype. The CD pairs are stored for each character position in the word. The SSS structure is combined with the CD pairs to predict the most likely character. The final decision of the choice of character is obtained by combining the results of the recognizer and the post-processor. Depending on how closely the character image matches with the nearest character prototype and the second nearest character prototype, a decision is made on how

much weightage has to be assigned to the post-processor and the recognizer. We have used two weights $w1$ and $w2$, where $w1$ is based on the distance of the character with its nearest matching prototype ($d1$) and represents the confidence of the recognizer. Smaller value of $w1$ means that the character image closely matches the library image and lesser weight has to be assigned to the post processor. The weight $w2$ represents the closeness of the shapes of the top two choices. Higher value of $w2$ means that the shapes of the character images of top two choices closely match and the confusion has to be resolved by assigning more weight to the post-processor. There is a quadratic growth of the weights with distance. These weights are combined with the percentages of occurrence of the top two choices at a particular position in the word and the distance of the top two choices with their nearest matching training set prototypes. The decision on the choice of one of the characters is taken as follows.

We calculate a parameter, dist, which represents the distance of the recognized character from the actual character. This has been formulated empirically as

$$\text{dist} = ((w1 + w2)/(100.0)) * (p2 - p1) - (d2 - d1)^2;$$

where $d1 =$ distance of first choice (char1) with the nearest matching library prototype,
$d2 =$ distance of second choice (char2) with the nearest matching library prototype
$w1 = d1^2$ subject to maximum of 50;
$w2 = 50 - (d2 - d1)^2$ subject to minimum of 0;
$p1 =$ percentage frequency of occurrence of char1 in char_freq_list;
$p2 =$ percentage frequency of occurrence of char2 in char_freq_list;

We recognize a character as char2, if dist > 0 else it is retained as char1.

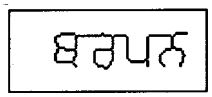
This is clarified with some examples:

Consider the following skeletonized images.

This image represents the word ਲੜ but it has been identified as ਲਤ by the recognizer, since the limbs of ੜ have been removed during the binarization and thinning stage. The CD pair generated for the third character is {(ੜ, 2), (ਤ, 6)}. The percentage frequency of occurrence of ੜ and ਤ in the third position in the corresponding SSS structure is found to be 0 and 90 respectively.

Thus values are $d1 = 2, d2 = 6, p1 = 0, p2 = 90, w1 = 4, w2 = 34$ and dist $= 18$ and since dist is positive so ਤ is replaced by ੜ

Using this technique we have been able to rectify more than one wrongly recognized character in a word. Consider the following example:



This image represents the word ਬਚਪਨ but it has been identified as ਬਰਪਨ by the recognizer. The CD pairs generated for the first two wrongly identified characters are {(ਬ, 3), (ਬ, 5)} and {(ਰ, 2), (ਚ, 6)}. The percentage frequency of occurrence of ਬ and ਬ in the first position in the corresponding SSS structure is found to be 0 and 100 respectively. Similarily the percentage frequency of occurrence of ਰ and ਚ in the second in the corresponding SSS structure is 0 and 100 respectively. Thus dist is found to have positive values for first and second character positions in the word and hence the best choices are replaced by the second best choices and the word is correctly identified as ਬਚਪਨ

A drawback of dictionaries is that a word, even if recognized correctly by OCR, finally gets replaced with some other word because it is non-standard and was not found in the dictionary. The same can be applied to abbreviations and acronyms. But in our current

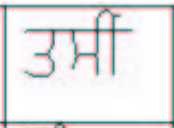
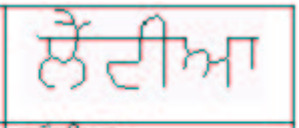
Word Image			
Recognized Word	ਪੱਚੀ	ਤਸੀਂ	ਲੱਦੀਆ
Actual Word	ਪੱਚੀ	ਤਸੀਂ	ਲੱਦੀਆ

Figure 3. Errors produced by merging/deletion of bindi characters in word images.

scheme a non-standard word, which is not present in the dictionary, but for which the OCR has a high confidence of recognition, will still be accepted and not replaced by the nearest matching word in the dictionary. For example, consider the character image used in the last example. If, say, the CD pair for the second character is {(ਰ, 1), (ਚ, 8)}, which implies that the character image very closely matches the character ਰ and not the next nearest matching character ਚ. The percentage frequency of occurrence of ਰ and ਚ in the second position in the corresponding SSS structure is 0 and 100 respectively. Then $w1 = 1, w2 = 1, p1 = 0, p2 = 100$ and dist $= -47$, which is a negative number and this implies that the character ਰ will be retained even though the word ਬਰਪਨ is not present in the dictionary.

6. Errors related to the upper dot

The character, ਂ (*bindi*), which is similar to a dot and is present in the upper zone was found to be responsible for a substantial number of errors. There were two types of errors produced

(1) *Deletion* – The character *bindi* would be removed during the scanning and binarization process or by the thinning algorithm. In many cases the *bindi* character would be merged with other symbols in the upper zone and vanish (figure 3).
(2) *Insertion* – The noise present in the upper zone would be confused with the *bindi*. Sometimes an upper zone vowel would be broken into smaller components, which would generate extra *bindi* characters (figure 4).

To tackle the *bindi*-related errors, we have used Punjabi grammar rules for the usage of the *bindi*. According to these rules, the *bindi* symbol should always be preceded by one of the following characters (ਅ ਆ ਇ ਈ ਸ ਹ ਕ ਖ ਗ ਘ ਙ ਚ ਛ ਜ ਝ ਞ ਟ ਠ ਡ ਢ ਣ ਤ ਥ ਦ ਧ ਨ ਪ ਫ ਬ ਭ ਮ ਯ ਰ ਲ ਵ ਸ਼ ਖ਼ ਗ਼ ਜ਼ ੜ ਫ਼ ਾ ਿ ੀ ੁ ੂ ੇ ੈ ੋ ੌ ੍ ੰ ੱ ੲ ੳ ਊ). We call this set of characters, set B1.

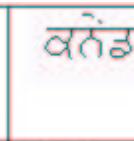
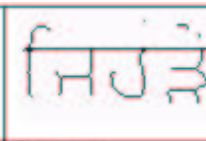
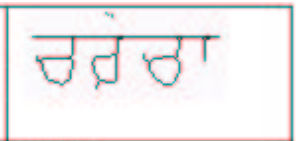
Word Image			
Recognized Word	ਕਨੰਡਾ	ਜਿਹ ੜੇ	ਚਰ ਚਾ
Actual Word	ਕਨੰਡਾ	ਜਿਹੜੇ	ਚਰਚਾ

Figure 4. Errors produced by extra generated bindi characters in word images.

The problem of filling the missing *bindi* characters could only be partially solved. If the recognized word is not present in the corpus, then we examine the positions in the word where one of the characters of set B1 is present and assumed that the *bindi* following that character has been accidentally eliminated and insert a *bindi* at that position. All the possible combinations are tried and the combination which yields a word with the highest frequency of occurrence in the corpus, replaces the word recognized by the recognizer. Using this methodology, we are able to rectify the error made by the recognizer in identifying the third word image of figure 3. The two *bindis* of the word were removed during the thinning stage and merged with the symbol in upper zone. The word was wrongly recognized as ਲੈਦੀਆ and it was found that there is no such word in the corpus. There are three characters of set B1 present in the word and so there are seven combinations of placing the *bindi* in the word (ਲੈਂਦੀਆ, ਲੈਦੀਂਆ, ਲੈਂਦੀਆਂ, ਲੈਂਦੀਂਆ, ਲੈਂਦੀਆਂ, ਲੈਦੀਂਆਂ, ਲੈਂਦੀਂਆਂ). All the combinations were tried and it was found that the word ਲੈਂਦੀਆਂ had the highest frequency of occurrence in the corpus and therefore it was selected. A limitation of this methodology is that if the recognized word is present in the corpus and if the *bindi* symbol is missing then it cannot be used. As for example both ਹਾ and ਹਾਂ are present in the corpus and ਹਾਂ has a much higher frequency of occurrence but we cannot convert any identified ਹਾ to ਹਾਂ as it may convert an actual ਹਾ into ਹਾਂ.

To weed out the extra *bindis*, the same grammar rule of Punjabi is again invoked. If a *bindi* is preceded by any character outside set B1, then the *bindi* can safely be removed. Using this strategy the extra *bindis* of the identified word ਚਰਾਂਚਾ of figure 4 are removed, since these *bindis* are preceded by character ਰ which is not a member of set B1. This technique will not erase the extra *bindi* produced by the broken upper vowel in word ਕਨੇਡਾ of figure 2 since here the *bindi* follows the character ` which is a member of set B1. For meeting these types of situations we again use the same technique that we used to pad in the missing *bindis*. If the recognized word is not found in the corpus, then it is an indication that it could be a *bindi*-related problem. We again try all the possible combinations of inserting *bindi* after a member of set B1 and the word with the highest frequency is accepted. Using this approach the image of word ਜਿਹੜੇ of figure 4 which is identified as ਜਿਹਂੜੇ by the recognizer, is corrected. Two superfluous *bindis* have been introduced in the word image. The first *bindi* is a noise speck while the second *bindi* is generated by the broken character`. The first *bindi* is eliminated using the grammar rule since it is trailing behind ਹ, which is a nonmember of set B1. The second *bindi* follows a member of the set B1 and so it cannot be deleted as such. But the modified word, ਜਿਹੜੇਂ , is still not present in the corpus. So it is checked if the new word minus the *bindi* is present in the corpus and the word ਜਿਹੜੇ , is present in the corpus and so the word image is identified as ਜਿਹੜੇ .

7. Application of grammar rules

Various Punjabi language grammar rules have been used to improve the recognition rate. These rules are mainly concerned with character combinations and valid starting characters in a word. The rules can be stated as follows:

(1) The first character cannot be a vowel or a half character or a member of the set (ਙ ਞ ਣ ੜ) (Set S1). The only exception is vowel ਾ. In case the starting character is a vowel or a half character then it is deleted, since the second best choice will again be a vowel or a half character, otherwise if it is a consonant belonging to the set S1 then it is replaced with the second best choice. If the second best choice is again a member of the set S1 then it is marked as an unrecognized character.

Table 2. Recognition accuracy of the OCR with and without the application of post processor.

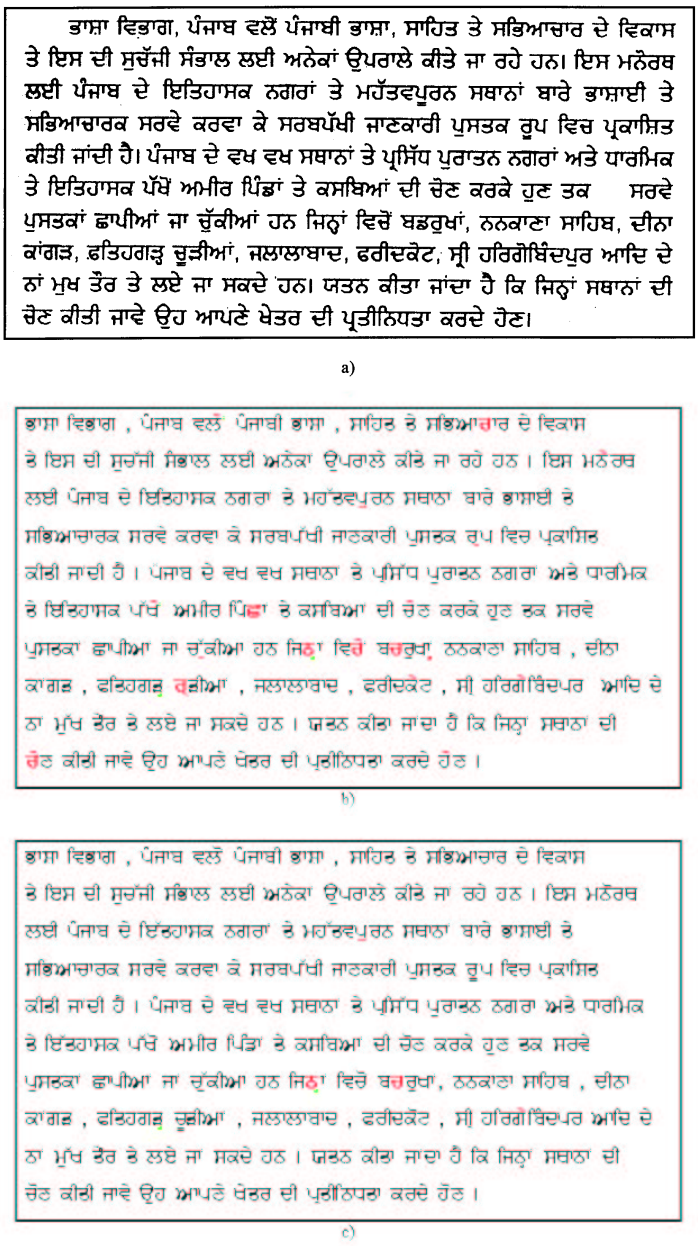
Zone	Recognition rate without post processing (%)	Recognition rate after post processing (%)
Upper zone	91.19	95.14
Middle zone	96.71	98.38
Lower zone	82.48	90.87
Overall	94.35	97.34

(2) A half character or a vowel in the lower zone cannot be immediately followed by another half character or a vowel in the lower zone. In such cases, the character which more closely matches the library prototype is retained and the other character is deleted. This type of problem occurs if a symbol in the lower zone is broken into two or more parts. Similarly, an upper zone vowel belonging to the set V1 (◌ਿ ◌ੀ ◌ੇ ◌ੈ ◌ੋ ◌ੌ ◌ੰ) cannot be succeeded by a member of set V1. If this happens, only the closest matching character is retained.
(3) The character ੳ should always be followed by the vowel ◌ੁ or ◌ੂ in the lower zone.
(4) The letter ਅ can never be followed by a member of the set V2 {◌ਾ ◌ੀ ◌ੁ ◌ੂ ◌ੇ ◌ੈ ◌ੋ ◌ੌ} . In case it happens the letter a is replaced by the next best option.
(5) The letter ੲ should always be followed by vowel◌ਾ or◌ੀ or preceded by letter ◌ਿ.In case it does not happen the letter ੲ is replaced by the next best option.
(6) The members of the set { ਙ ਞ ਣ} cannot be followed by a member of the set {◌ੰ ◌ਂ}. In case it happens the member of the set { ਙ ਞ ਣ} is replaced by the next best choice.
(7) The half character◌੍ should always be preceded by a member of the set {ਨ ਮ ਵ ਰ ਲ ੜ ਸ} otherwise it is replaced by the next best choice.
(8) The half character◌੍ should always be preceded by a member of the set {ਸ ਦ ਧ} otherwise it is replaced by the next best choice.
(9) The half character◌੍ should always be preceded by a member of the set { ਸ ਮ ਸ਼ ਕ ਪ ਗ ਘ ਟ ਬ ਤ ਦ ਧ◌ਾ◌ੀ◌ੇ◌ੈ◌ੋ◌ੌ ਾ ਭ}. In case it does not happen, the character which is more closely matches its prototype is left unchanged and the other character is replaced by the next best choice.

The decision regarding the character to be replaced, in case of an invalid character combination, is made by looking at the past history of the recognition rate of the recognizer for the character. For example, in rule (4) we have replaced the character ਅ with the next best option in case of an invalid character combination instead of the member of set V2 as it was observed that the chances of wrongly recognizing a character of set V2 are very low. There could be confusion between the members of set V2 but it was rare that a non member of set V2 was recognized as a member of set V2 or vice-versa.

8. Experimental results

We have tested the performance of the OCR and the post processor on a large variety of text images which included scanned images from books and laser print outs. The images were scanned at 300 dpi resolution. The performance of the post processor on a text image is given in figure 5. The mis-recognized characters are drawn in red and the missing characters are redrawn in green.



Note – Red text represents wrongly identified character; Green text represents missing character in the recognized text

Figure 5. **(a)** A sample text image. **(b)** Recognized text with post-processing. **(c)** Recognized text after post-processing.

The recognition accuracy of the OCR without post processing was 94.35%, which was increased to 97.34% on applying the post-processor to the recognized text. The recognition rates for the characters in all the three zones are tabulated in table 2.

Figure 6. Word images with confusing character images of ` and ~.

Some other observations which were made during the experiments are:

(1) The upper zone vowels (` and ~), because of the similarity in shapes and large possible shape combinations (figure 6) were greatly confused by the recognizer and the confusion was partially resolved by the post-processor. The other source of error in the upper zone is the character *bindi*, which has already been discussed in one of the previous sections. As can be observed in table 1, the recognition accuracy of upper zone characters is 91.19% which is improved upto 95.14% by the post-processor.
(2) The recognizer performed very well on the characters in the middle zone, which is the busiest zone. A majority of the errors in this zone were made in the recognition of the visually similar character pairs (ਬ and ਥ) and (ਚ and ਰ). The recognition rate of middle characters is 96.71%, which is further improved to 98.38% by the post-processor.
(3) The lower zone, in which 2 vowels and 3 half characters reside, accounts for approximately 3.5% of total character population, proved to be the most difficult zone to remove errors. Some of the causes of the poor performance of the recognizer are the similarity in shapes of the characters, small size of the characters and merging of the lower zone characters with the middle zone characters. The recognition rate of the lower zone characters was observed to be 82.48%, which has been improved to 90.87% by the post-processor.

9. Conclusions

A post-processor for improving recognition accuracy of OCR for Gurmukhi script has been developed. It uses information derived by analysing an available corpus of Punjabi text. The corpus is used to obtain frequency of occurrence of different characters, at specific positions in a word. In addition, the corpus is used to derive information that is useful for disambiguating visually similar-looking words. Based on the values of some simple binary features, the character set is partitioned into subsets of characters that are visually very similar. Using statistics of such similar characters occurring in different positions in the word, a novel strategy is employed to correct character recognition errors. In addition Punjabi language rules and heuristics for upper dot are utilized. A 3% increase in recognition accuracy from 94.35% to 97.34% has been reported on clean images. This is the first time that a post-processor has been developed for Gurmukhi script OCR but it has been tested only on clean images. The development of a post-processor for degraded images is a topic for future research.

References

Bansal V, Sinha R M K 1999 Partitioning and searching dictionary for correction of optically read Devnagri character strings. In *Proceedings Fifth International Conference on Document Analysis and Recognition* (IEEE Comput. Soc. Press) pp 653–656

Church K W, Gale W, Hank P, Hindle D 1990 Word association norms, mutual information and lexicography. *Comput. Linguistics* 16: 22–29

Hong T 1995 *Degraded text recognition using visual and linguistic context*. Ph D thesis, Faculty of Graduate School, State University of New York, Buffalo, NY

Hull J J, Srihari S N 1982 Experiments in text recognition with binary n-gram and Viterbi algorithm. *IEEE Trans. Pattern Anal. Machine Intell.* 4: 520–530

Lehal G S, Chandan Singh 1999 Feature extraction and classification for OCR of Gurmukhi script. *Vivek* 12: 2–12

Lehal G S, Chandan Singh 2000 A Gurmukhi script recognition system. In *Proceedings 15th International Conference on Pattern Recogniton*, Barcelona, Spain, vol 2, pp 557–560

Mayes E, Dameran F J, Mercer R L 1991 Context based spelling correction. *Inf. Process. Manage.* 27: 517–522

Riseman E M, Hanson A R 1974 A contextual postprocessing system for error correction using binary n-grams. *IEEE Trans. Comput.* C-23: 480–493

Sinha R M K 1987 Rule based contextual post-processing for Devanagri text recognition. *Pattern Recog.* 20: 475–485

Suen C Y 1979 N-gram statistics for natural language understanding and text processing. *IEEE Trans. Pattern Anal. Machine Intell.* 1: 164–172

Tong X, Evans D A 1996 A statistical approach to automatic OCR error correction in context. *Proceedings of the 4th Workshop on Very Large Corpora*, pp. 88–100

Wells C J, Evett L J, Whitby P E, Whitrow R J 1990 Fast dictionary lookup for contextual word recognition. *Pattern Recogn.* 23: 501–508

Yannakoudakis E J, Tsomokos I, Hutton P J 1990 N-grams and their implication to natural language understanding. *Pattern Recogn.* 23: 509–528