

Designing a Front End OCR System for Indian Scripts for Machine Translation - A Case Study for Devnagari

Veena Bansal
veena@iitk.ernet.in

R. M. K. Sinha
rmk@iitk.ernet.in

Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur 208 016 India

Abstract

An OCR system constitutes a natural way of providing input to a machine translation system. The document is optically scanned. The front end OCR system extracts text zone and reads the text. The translator, when presented with recognized source text, produces a corresponding document with the text content translated in the specified language.

Indian scripts are two dimensional composition of symbols. Therefore, a text reading system, first segments the text into symbols. The segmented units are classified with the help of prototypes for the symbols. The words are composed back from the output of the recognition process. Words are verified with the help of a word dictionary. It is at this stage, the interface to the translation system is of great help. All translation systems have a dictionary and a sentence analyzer. A sentence analyzer, in the form of a formal grammar parser, will reject sentences where substitution errors have occurred except in those cases where the substitution errors lead to other valid words of the same syntactic category. A sentence analyzer which is expectation driven, provides clues for alternative words in case of substitution errors.

INTRODUCTION

The optical machine reading of text, besides providing a keyboardless interface, has a variety of commercial and practical applications in reading forms, manuscripts and their archival etc. It is a great help to the visually handicapped when interfaced with a voice synthesizer. It may be used as a front end to a machine translator. A document consisting of text and images is optically scanned. The front end extracts the text zones from the document. The extracted text is segmented into characters. These characters are classified into known classes with the help of prototypes [?], [?]. The prototypes are obtained during a separate training phase. A document image and the corresponding true text is provided to the trainer. It is the responsibility of

the trainer to segment the text image into characters. The trainer must also establish a correspondence between segmented character images and the true characters. The prototypes obtained during training phase are used by recognition process to estimate the degree of match with an unknown symbol. In case, an unknown character is classified in a single class with high confidence, no further processing is required. However, a character may not be classified to a known class resulting in a rejection error; to a wrong class leading to a substitution error. The extent to which a system generates 'rejects' or 'substitution errors' is dependent upon the threshold used at the classification stage. A substitution error cannot be detected at the character level. A word level knowledge is required to correct rejection and substitution errors.

A translation system dictionary contains additional information regarding each word. Detailed information about the syntactic category and some semantic information are included with each word. An OCR dictionary contains no information about the word. We make use of the additional information available while checking the presence of a word in the dictionary. A word mapped to another dictionary word is detected if the syntactic category is different from what the sentence analyser expects. In case, the syntactic category remains unchanged, the semantics are checked. Before going into the details of the OCR system, let us examine the distinctive features of Indian scripts which make the problem more challenging.

SOME RELEVANT FEATURES OF INDIAN SCRIPTS IN GENERAL AND DEVNAGARI IN PARTICULAR

Indian scripts are different from Roman script in several ways.

- Indian scripts are two dimensional composition of symbols, core characters in the middle strip, optional modifiers above and/or below core characters..
- All Indian scripts are alphabetic unlike ideographic scripts like Chinese.
- A character may have many different forms and the script specified rules for their usage in different context.
- Two characters are required to touch in some cases.
- Two characters may be in shadow of each other due to their shapes and writing styles.

Devnagari script has 40 characters which can be written as individual symbols in a word. Most of the characters have a *half* form which forms a conjunct with the following character. Some characters of Devnagari script take the next character in their shadow because of their shape. The script has a set of modifier symbols which

- (a) Words showing header lines:
- (b) Words with lower modifiers:
- (c) Words with shadow characters
- (d) Words with composite characters:
- (e) Words with rkar modifiers:
- (f) Characters of different height and width:

Figure 1: Examples from Devnagari Script

are used only to modify a character. Every word has a header line placed above the core characters. Top modifiers are placed above the header line. The lower modifiers are placed below core characters. More than one lower modifier may also be placed below one character. A character may be in shadow of another character, either due to a lower modifier or due to the shapes of two adjacent characters. Some examples are shown in figure ??.

Above examples give an idea of the complexity of the script and various ways in which a character takes form of a composite character.

STRATEGIES FOR READING INDIAN SCRIPT TEXTS

It is apparent from the above discussion that a word of Indian script needs to be first segmented into composite characters and then each composite character needs to be decomposed into a set of symbols. These symbols may correspond to a Devnagari character, a modifier symbol or may not have any meaning when viewed in isolation (this normally happens when the characters get fragmented or a modifier symbol extends from upper to core-strip region of the word). It is these decomposed symbols for which the trainer constructs the prototypes for the recognition stage. These symbols are composed back into characters which are used to obtain valid words. It may be noted that the composition process is specific to Indian scripts and there is no such composition required for Roman OCR. The symbols which do not produce valid words are candidates for rejection and substitution errors. Some of these errors are corrected with the help of composition rules of the script [?]. Positional information of symbol boxes is used to associate boxes with the core character/symbol. There is

room for ambiguity in composing the symbols. However, the composition rules help in proper association formation.

MULTIPLE SCRIPT PROBLEMS

It is becoming commonplace to use English words in an Indian script document. The recognition process may first identify the script and then invoke corresponding recognition module. Features of the scripts such as header line, placement of characters in the text line, aspect ratio etc. may be used to identify the script. The other approach is to augment the set of the Indian script by Roman script [?]. This approach increases the size of a class substantially which may require more features for unique classification. However, this strategy fails with Indian scripts due to the nature of segmentation required. Consider Roman letter 'T', the horizontal line resembles header line of Devnagari script word. The segmentation process without the knowledge of the script, will remove the horizontal line and the resulting character will be classified as 'I' which is a character in Devnagari script.

SEGMENTATION AND DECOMPOSITION

Due to two dimensional composition of Devnagari script, the segmentation is done in two phases. For initial phase, the horizontal and vertical histogram form the main basis for segmentation. The second phase or the refinement phase uses the statistical information about height and width of units obtained in the first phase. The refinement process makes use of structural properties of the script. The segmentation of uniform text zones into lines; lines into words; words into recognition units all are done following the same strategy.

Uniform text zone is segmented into text lines with the help of horizontal histograms. The overlapping text lines are handled by second phase that uses an estimate of the line height from the lines obtained in the first phase.

Identification of word boundaries in Devnagari script is straight forward due to the header line that joins all core characters of a word together. The characters and symbols of a word in Devnagari are joined together by header line and a word is vertically separate from the following word. Using these features, a text line is segmented into words. Segmentation of a word into constituent recognition units requires identification and removal of the header line of the word. However, header line is easily located being the most dominating horizontal line in a word. Header line is removed from the word and the zones above and below header line are treated separately. Notice that all upper modifiers are in the zone above the header line. Whereas, the lower modifiers are still clubbed with core characters. Devnagari script

has characters of varying height; a character taller than the rest in a word may not have a lower modifier. Therefore, we first extract characters and symbols that are vertically separate from their neighbours. Width and height of these extracted units is analyzed for the most frequent height and width.

A text page written in Hindi which uses Devnagari script has less than 20% composite characters which includes shadow characters and conjuncts. This percentage is found in a large number of randomly selected text pages. We make use of this observation to cluster the characters in three different bins based on their width. The characters which are wider than the most frequent width are candidates for further segmentation. Further segmentation is not attempted until a suspected composite character is either rejected by the recognition processor or is classified with low confidence. Since a composite character involves two full characters or one half and one full character, it is unlikely that a composite character will get mapped to another character with high degree of merit. The characters with lower modifiers are also about 20% in a text page. This percentage holds for a text line, though it doesn't hold for a word. The most frequent height is found from the height of units obtained from preliminary segmentation. All characters that are taller than the frequent height are likely to have a lower modifier.

The second phase of the segmentation is invoked after seeing the output of the classification process. Suspected composite characters are decomposed using heuristics and structural properties of the script.

SYMBOL RECOGNIZER

After the word is segmented into characters and symbols, an attempt is made to classify each symbol into known classes. We use a hybrid approach for classification. Devnagari characters is divided into three classes based on the presence and position of vertical bar of same height as the character. These classes are:

No bar characters

Middle bar characters

End bar characters

This feature is visually extracted and provided to the recognition process. Initially, the complete Devnagari set is the set of candidate characters for an unknown character. Vertical bar property is used as filter to eliminate some of the characters from candidate set. Horizontal zero crossing is also used as a filter to reduce the candidate set further. In a single known font environment, moments of degree up to 2 and aspect ratio of characters are also used for classification. Prototypes for all these

features are extracted during a separate training phase. In a multifont environment, statistical features are not used due to their sensitivity to font type. Instead structural prototypes are used for classification. The structural descriptions are also generated by an automated process during the training phase. Top three choices are retained for composition and verification. Our experiment has shown that 95% time, the true character lies in the top three choices.

A TWO-PASS TRAINING SCHEMA

In order to train the recognizer, prototypes are constructed for the known samples. We need to construct a file of "true" data which correspond to the sample text. While for Roman, it is simply an ASCII file which represent the individual characters of Roman script, in Devnagari this file is in the form of ISCII (Indian Script Code for Information Interchange). However, the ISCII codes do not correspond to the symbols as obtained through the process of segmentation and decomposition as outlined earlier. The training process should present the unknown symbols in the same form and order as obtained during the process of recognition rather than the idealized symbols.

For automating the training process for construction of prototypes for Devnagari, we employ a two-pass schema. In the first pass, the trainer as usual extracts the lines from the text and finds the word boundaries. If the number of words in the input text does not match with the expected number of words as per the true file, the whole line is ignored for training during this pass. Preliminary segmentation process yields us a number of physical symbol boxes corresponding to each word. At this stage if there is no mismatch in the number of symbol boxes as obtained physically and the expected number derived from the true file, information about width and height of the characters / symbols are collected and analyzed statistically for use in the second pass. In addition, the prototypes are constructed for these symbols.

In the second pass, we concentrate on those words which have earlier shown a mismatch on the number of symbol boxes. A number of heuristics are used to ascertain the plausible fused or shadowed boxes. Similarly, after the segmentation also, a number of constraints are applied before accepting the box correspondences for the training purpose.

The Training Process

Figure ?? presents an illustration of the process of training for construction of prototypes for Devnagari.

The trainer has two inputs: the scanned input data and the true-file text internally represented in ISCII code which is obtained after linearization of characters

- (a) Input text words:

- (b) Sample true-file symbols as represented by ISCII code (Phonetic order):

- (c) Physical symbol boxes obtained through optical processing of words of (a) assuming no fusion or fragmentation:
 - Upper region:
 - Core-strip :
 - Lower region: Γ

- (d) Rearrangement of symbols of (b) into visual-order after applying composition rules:
 - (Composition rules used in the example: = , =)

- (e) Delineation of different regions of (d) above:
 - Upper region:
 - Core-strip :
 - Lower region: Γ

Figure 2: An Example illustrating the training process: (a) Input text words; (b) Sample true-file symbols as represented by ISCII code (Phonetic order); (c) to (e) show the stages through which the true file needs to be transformed to provide one-to-one correspondence of symbols for training. The example is under the assumption that there are no fusions or fragmentations.

and symbols in the phonetic order [Sinha 84] [Sinha 92]. These correspond to as given in figure 5(a) and 5(b) respectively. As outlined earlier, the input scanned data is segmented into lines and lines into words. Thereafter each word is attempted to be segmented and decomposed into individual symbols [?]. This process is hindered by character shadowing, character fusion and character fragmentation. Hence the number of physical symbol boxes may not correspond to the expected number of boxes. However, if there are no fusions or segmentations, then the symbol boxes obtained for the example are as shown in figure 5(c). As shown the symbol boxes belong to three strips called upper modifier region, core-strip region and lower modifier region [Sinha 79]. The arrangement of the symbol boxes is that of visual order. The task of the trainer now is to establish correspondence between these physical symbol boxes (figure 5(c)) with those which correspond to true file (figure 5(b)). Firstly, the true file which is in phonetic order of symbols for internal representation, is rearranged to visual order using a set of rules for transformation [Sinha 84] and script composition rules [Sinha 87]. This is shown in figure 5(d). Thereafter, this is further delineated into the upper modifier, core-strip and lower modifier regions as shown in figure 5(e). Now we observe that there is one to one correspondence with the OCR physical symbol boxes as shown in figure 5(c) and those of figure 5(e) obtained from the true file. The process of construction of prototypes can begin now.

COMPOSITION OF RECOGNIZED SYMBOLS INTO WORDS

This module composes recognized characters and symbols back into valid Devnagari words. A set of rules defining the script composition is used. Symbols which do not form legal words are corrected if possible with the help of composition rules. More than one valid word may be formed as an output of this stage with alternatives provided by character recognition process for some of the symbols/chs of the word.

INTEGRATION WITH TRANSLATION SYSTEM

All valid Devnagari words obtained after the composition process, are validated through a dictionary. A dictionary for OCR contains a word with no additional information about the word. A word has many forms in Hindi language as in case of English. Dictionary may contain just the root word for different forms of a word or may contain all possible forms of the word. Size of dictionary is about 20,000 words. These words are usually partitioned for fast search. The partitioning strategy is such that that the right partition is searched inspite of certain recognition errors in the word. A word found in the dictionary is assumed to be correctly recognized. Sometimes, a substitution is required for mapping a word to a dictionary word. Consider

There are two errors in this sentence that are easily corrected by consulting the dictionary. Word is mapped to and to . However, there is a possibility that the input word is mapped to another dictionary word due to classification errors. We have two such examples from the real text and testing:

Input1:

Output1:

Input2:

Output2:

In both cases fourth word is mapped to another dictionary word. With no additional knowledge about words and their usage, no further enhancement in the performance is possible. If the system is interfaced with a machine translation system, additional information about each word is available.

These errors can be detected and corrected by using additional information that a lexicon dictionary has. A lexicon dictionary has syntactic category of a word and, syntactic and optional semantic expectations for the following word/phrase. A verb entry includes its type and usage, type of subject and object(s) for the verb. A noun is described by its gender, number and person. A class is also associated with each noun such as *thing, animate, picture, machine* etc. A word mapped to another word is easily detected if the syntactic category of the mapped word is different from the expectations generated by previous word. In case, syntactic category remains unchanged, other details like gender, person, number and class is checked. The sentence analyzer of the machine translation system has the functionality required for verification. An expectation driven sentence analyzer generates expectations and checks the words produced by OCR. The words which meet the expectations are accepted. For example 1, the verb doesn't expect an animate object unless it has present, i.e. . Therefore, the dictionary is searched for an alternate word. An alternative word from the dictionary is selected in a constrained manner. These constraints come from width of constituent characters and words. During testing phase, output of the classification process is used to learn confusions for various characters. A match is preferred if substitutions are done with the confusions of input characters. For our examples, the confusion matrix is used and both the errors were corrected.

The above discussion may lead a reader to believe that a word is always mapped to a dictionary word. This is not the case; inspite of enormous size of the

dictionary, proper nouns are usually excluded. Foreign words, acronyms or domain specific words are likely to be absent from the dictionary. The verification process is not allowed to map a word to a dictionary word blindly. Therefore, the confidence level associated with each character of the word is used to allow substitution. The characters which have been recognized with low confidence are the only ones allowed to be substituted.

CONCLUSIONS

Communication is becoming easier and faster everyday. No one can afford to be isolated. The work must be available on network in electronic form. An OCR interfaced with a machine translator is a system that can take a document and produce corresponding electronic version in a specified language. In this paper, we have addressed the issues involved and suggested some of the solution when Hindi is the input language. The dictionary of a machine translator and sentence analyzer are used to verify the output of OCR. Some of the errors which an OCR system would fail on, have been corrected because of the interfacing with translation system. Therefore, it is not just a producer-consumer interface; the interface helps OCR in enhancement of its performance.

References

- [1] R.M.K.Sinha and H.N.Mahabala (1979), Machine recognition of Devnagari script, *IEEE Trans. on Systems, Man and Cybernetics*, SMC-9, 435-441.
- [2] R.M.K.Sinha and Veena Bansal (1995), On Devnagari Document Processing, *IEEE International Conference on Systems, Man and Cybernetics*, Vancouver, Canada 1995.
- [3] Veena Bansal and R.M.K.Sinha (1996), On integrating diverse knowledge sources in optical reading of Devnagari script, *Proceedings of the international conference on Information, Systems, Analysis and Synthesis*, ISAS'96, at Orlando, USA.
- [4] R.M.K.Sinha (1987), Rule based contextual post-processing for Devnagari text recognition, *Pattern Recognition*, 20, 475-485.
- [5] R.M.K. Sinha (1984), Computer processing of Indian languages and scripts - potentialities and problems, *Jour. of Inst. Electron. & Telecom. Engrs.*, vol.30,no.6, 133-49.