On Developing OCR Software for Tamil

Dr V Krishnamoorthy
(Former Professor Of Anna University) INFOREED
11, Fourth Street, Padmanabha Nagar, Adayar, Chennai 600 020 profvk@softhome.net

---

ABSTRACT

We are developing a software for optical character recognition for printed Tamil text. A novel method of representing a character as a graph is used. The graph is generated in parts, without using any thinning algorithm. This saves time. Though we loose much information in this method, the information got is sufficient to recognize the character. Some of the problems we encountered in developing this software are presented here. Some possible approaches to solve these problems are also discussed.

INTRODUCTION

We start the discussion by first specifying the method of approach. There are many approaches in designing OCR. We have chosen a new method, based on representing a letter as a graph. If we consider a letter as strokes made up of lines, it can be considered as a graph, by inserting a few vertices at the required places. These vertices can be the end points, points which are local minimum or local maximum in the x and y directions. This representation of a letter as a graph has some advantages and some disadvantages.

The major advantage is that the information content gets reduced very much. Hence the processing needed to recognize a character gets reduced very much, in many cases. This speeds up the recognition process very much. We have devised a method in which this graph is constructed without thinning the character map. This again adds to the speeding up of the recognition process.

The major disadvantage is that some times too much information gets lost, and we have to resort to different methods to recognize a letter.

LOOK ALIKE LETTERS

On problem which is common to all the algorithms is the set of letters which look very much the same, except for some minor differences. For example, in Tamil, the letters ka and su differ only at the end of the strokes. This is because in many of the fonts we see the end of the letter ka does not touch the body and hangs loose.

The letters la and va give the maximum trouble. The bitmaps for these two letters differ only in very few bits. Due to some noise this difference is not that much pronounced in some cases. Here it becomes very difficult to distinguish between a la and a va. As a first step, we look at the original bit maps more closely and try to distinguish between them. This also may not work satisfactorily many times. This is due to the fact that we always give some allowances for

eliminating noise.

We can try different methods to guess the letter in the following way. By looking at the previous letter we can decide this character. In Tamil, we know that only certain characters can follow certain characters. But this may not work in all the situations.

As an extension of this method, we can look at all the other letters in that word and then try to decide. If some other characters in this word also could not be decided this becomes more complicated. Again a dictionary of words have to be used for this purpose. But this is not a problem. The real problem is that the dictionary cannot be used as such. In Tamil, we have words formed by combining many parts. It may be surprising to note that as many as 13 parts can be within a single word. For example, paarththukkondirundhavarkalaiyumthaneyada is such a word. Though this type of words may not be common, many words contain around 3 parts. Hence it is not just a simple table lookup, or just a simple modification of a table lookup.

This problem is similar to the problem of suggesting alternatives by a spellchecker, when a word is found to be wrong. Some Tamil word processing software have tried this. The approach in these software seems to be matching only the first few characters. They do not seem to bother about what happens at the right side of the word. As such sometimes we may see some unexpected results. Also they may not suggest anything if the word consists of many parts and the error occurs near the end.

Our problem here is a bit simpler than the above one. If we could have a spellchecker, then by testing all the combinations (by changing la into va etc.) we could end up with the correct one. This is again when more than one is not a correct word. Some of these combinations could be eliminated based on the previous discussions on adjacent characters.

If special methods to tackle the pairs like la, va could be found based on the linguistics it would be very useful.

SPLITTING OF LETTERS

While splitting the letters from the adjacent letters, kerning have to be taken care of. Usually the ending in the letter tha goes below the previous letter. This problem becomes acute when we deal with fonts which are slanting in their normal shape itself. We face more problems with the slanting letters. If we try to straighten them first, more noise is introduced due to the digital nature of the pixels. If we keep them in their original shape, even splitting them into letters have to be handled differently.

Touching of adjacent letters

Another problem we face is the joining of two adjacent letters into one unit due to spreading of ink. Sometimes it may be recognized as a different letter. In that case nothing could be done. But when it does not match with any known letter, then it is a challenge to find the two characters separately. Since the width of the unit to be recognized is a bit longer, it may be possible to guess that it consists of two characters. But to cut them into two is not easy.

A method may be devised in which we try to reduce the search space of the first character just by seeing only a portion of the left side of the character. Similar exercise may be made for the right portion of the second character.

DISTORTIONS

One more problem we face is the following. In order to ignore the distortions created by spreading of the ink, we have to omit some odd dots. Similarly we omit some white space and put some dots. This when combined with the ink spreading or not spreading, increases the distortion. The letters na (rannakaram) and la (as in the plural ending kal) give trouble in this way. The loop in the la disappears and forms one line. This results in wrong or non recognition. In such cases some other method like neural network may be tried.

OTHER ISSUES

The following situations are not handled for the present. This is because we are trying to research and see whether our OCR software will work first satisfactorily. They can be taken up only at a later stage.

1. Same line has more than one font.
2. Normal and italics are mixed.
3. Same line has fonts of different sizes. Usually the first letter of a paragraph has a bigger size letter.
4. Some letters are underlined.
5. Some pictures are in the page.
6. More than one column is printed.

CONCLUSION

We have highlighted some problems involved in developing an OCR for Tamil printed text. We have suggested some methods for tackling the same. All the mentioned solutions are complicated. They will increase the time for recognition very much. But, if the accuracy required is more than 99%, it may be necessary to employ all these techniques. Also, the logic should be built in such a way that each situation is handled intelligently so that the time taken is small. The shape of the characters, the linguistic nature of the words, and the different approaches for character recognition- all these have to be mixed and used judiciously to get these best results in OCR. The recognition process should stop at the earliest. Only when required, based on the particular situation, the required special checking should be taken up.