Challenges faced in developing handwriting recognition software for Tamil

Dr. V. Krishnamoorthy
(Former Professor, Anna University)
Learn Fun Systems <prof.vkrish@gmail.com>

ABSTRACT

The third version of our online handwriting recognition software is coming out now. It takes care of almost all normal variations in which a character is written, and also some large variations. It needs no training is a point to note. Here we trace the problems faced, the decisions taken, the progress made and the problems remaining in each version. The problems faced include the variations in writing one character, similarities between many characters, the extra tails at the beginning and end, the relative sizes between the parts of a character, the various screen densities, the holding of the device influencing the direction of writing. Though our basic method remains the same for many languages in which the letters are written individually, our experience shows that each language and each letter of a language has to be treated individually. Required data and algorithms have to be created separately for each language and the generic engine will take care of the prediction.

Introduction

Though speech is the easiest form of communication, the other two commonly used forms, namely writing and typing will have their own importance in different contexts. Tapping on the keys on the touch screen comes under typing. Swyping can be considered as writing in a new script. Morse code like methods are specialized, and are not used by the common man.

When the computers became a household article, its use by the common man in India in his local language was not widespread, mainly due to the large number of characters to be typed. With the advent of the cheap touch screen mobile phones and tablets, a new era has dawned in communication by the common man in India. With the availability of the online handwriting technology, now it is possible for

anyone to get the benefits of the computing technology, like getting information available on the net and easy collection of data at the site. Even the elementary school children can search the net with this technology, without learning to type.

First version

Realizing the importance of this technology I started working in this area in 1994. Our software with the first version of online handwriting recognition for Tamil came out a few years ago. Touch screens were not commonly used at that time. We actually develop our software by writing on the screen using a mouse, in the Windows platform. Writing using a mouse will usually create bad looking letters. But we were happy because we could recognize even these letters.

The first version, as with any technology, was more of a proof of concept. Many things had to be considered even for the first version. It has to tolerate some variations in writing, if not for large variations. A major decision had to be whether to have training sessions or not, for individual users. If yes, then is it going to be for all the letters or only for some letters? The technology we were developing took care of most of the variations in writing a letter. The data created had all the information. This data is kept in specialized form to save the time in processing. It would be very difficult to include these data from the samples got from the user in the training sessions. Also, most of the people would like to start using a software without any training. Simply put, people do not like training sessions. Hence we decided that our software will not have any training module. But this puts a heavy Burdon on us. We have to cater to all variations at the compilation time itself.

The first version was available in PonMadal, and in one version of PonMozhi. It took care of many variations in writing. For example, the letter 'thi', can be written with one, two, three or four strokes. The letter 'tha' can end at the bottom left or bottom middle or bottom right. That is, anywhere in the bottom.

When a letter is written, at the touching of the finger on the screen, there may come a small part of the stroke which comes in a direction which is not intended. Also at the end sometimes a letter has an extra bit going in a different direction. This is sometimes due to the way the hand moves to start the next letter. These two extra bits poses major problems to the designer. The dilemma is whether to cut something or not, and if yes, how much is OK. The problem becomes more acute when we have devices with small number of pixels per inch to large number of pixels per inch. We have to cater to all the devices. Also, the user may write in many sizes. Design decisions had to be taken based on our experience. They may work well in most cases, but give problems in some cases. There is no one best solution.

There is another problem in the way a letter is written. For example, the letter 'pa' has to start from the top. But many people start this letter from the bottom, go to the top, and then retrace this path to the bottom. This extra bit is not small, as the one we mentioned in the previous paragraphs. This bit spans the entire height of the character. These variations had to be considered separately.

We decided that we will try to fix one best matching letter first. Then two or three similar looking letters will be provided as alternatives. Though this worked well in most places, it failed in some places. Minute variations between similar looking letters forced these errors. The following is a list of some similar looking letters.

Another major challenge is when a letter is written using many strokes. The problem is in deciding which of the strokes are to be combined to form a letter. 'Intersecting strokes are to be combined' - is a simple solution, but not a complete one. For example, 'poo' can be easily confused with 'ta too'. The ordering of the pullis forming the letter 'ak', and differentiating it from three dots is a challenge. The problem is that even six non intersecting parts can form one letter. For example, see the letter 'gnu' shown below.

The pulli for a pure consonant seems to be a simple thing. But it poses major problems. Pulli is written in many sizes. Some write is so big that it can be confused

with the kokki for ikaram or eekaaram. Also the pulli may not be exactly above a letter. All these have to be taken care of. A kokki written slightly to the right of a character can be confused with the thunai ezhuththu.

Some ambiguity can be resolved by post processing. For example, if a letter is not the first letter of a word, then between the vowel 'ae' and single kombu, it is single kombu, since in Tamil, usually a vowel does not come in the middle of a word. Though we have a fast spell checker, we decided against using it, because of two reasons. First, this can be used only when a word is completed. Since we wish to get the letters correct immediately after it is written, we do not want a post processing after a word. Second, this may change the word incorrectly when words which are not usually in the dictionary, like the names, are written.

Second version

After a while we realized that though the technology we had developed works, it can be modified and made less complex. This will make it work better and a bit easier to implement. Hence a major overhaul of the design was undertaken. We redesigned the way we look at a letter. We simplified our approach. This simplified our data structure. We included more variations for some letters. The second version was born. It was faster also.

The problems started when people started expecting a perfect software. When people tested this software, they expected that it should recognize in whatever way they write. Some wrote very fast as if they were in a hurry, and hence some parts of the strokes were missing. Some scribbled as if they were signing. Some wanted to see how much variation it can tolerate. Some wrote in very small sizes. New ways in which some letters are written were discovered. For example, the letters like 'thunai ezhuththu' and 'ka' are written as follows. We had not taken care of this variation before.

At this stage we thought that we have to do a better job so that it will satisfy most, if not all.

Third version

The work for this version started about one and a half years ago. Our main goal was that we should recognize whatever is written. Do not leave anything. Predict more than one letter. Allow for large variations, even though this may clash with other letters. Include many more forms of writing, whether someone writes like that or not.

Do not leave anything as unrecognized - This is the philosophy with which we started our work for the third version. Hence large variations in different parts of a letter had to be allowed. This meant that we will have many predictions for a letter. We had to do a better job in identifying which one is better than the others. Hence we had to produce a number to grade them. We had to involve this grading while evaluating every feature of a letter. Our data structures and algorithms had to be modified thoroughly. We decided that we will provide up to four predictions per letter. They will be according to the order in which our procedure rates them. The first one with highest ranking goes to form a word. The others will be shown alongside. If the user wants, he can choose an alternative.

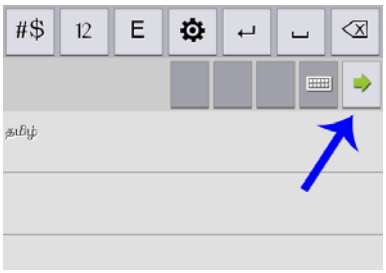
After making all the modifications, we found that the recognition engine has slowed down considerably, when used in mobile phones. Now we had to concentrate on the way the algorithms work, and find ways of speeding up, without compromising on the quality of recognition. This took some time, but we succeeded. All along we have tried to keep the size small. Many are surprised when we say that the core engine for recognition is just about 300KB per language!

One problem which remains is that some people write in a direction pointing to the top right. This makes the letters slant too much. Though some slanting is taken care of, abnormal slanting poses a problem. Now the third version is available in our software for Windows, Android and iOS.

From our experience with this technology we found that our core technology can be used for all Indian languages. But for each language, the data and functions have to be tailored to suit that language. The major consideration has to be in differentiating

between similar looking letters. We have started porting this third version of our technology to other Indian languages. It will work for all languages where the letters are written individually.

Searching the net for handwriting technology will show only a handful of companies offering this technology. This is due to the complexity involved. Finally, the question is whether everybody's handwriting can be recognized. Our view is that though most of the time we can recognize the letters, there may be a few occasions where the software fails. These may be due to some big deviation in some portions of a letter, or due to some software error. Our aim is to provide software which can recognize any normal or near normal writing. Towards this goal, we have included a provision to capture the points in the strokes. If a letter is not recognized, and the user thinks that it is written normally or near normally, he can long press the green right arrow button provided in the row where the recognition and options are shown. The data thus collected will help us improve the recognition capability of this software. For anyone who is willing to optimize this software for his or her handwriting, we will be happy to send a copy of the Android test version.



Conclusion

We have traced the challenges we faced while creating the technology for handwriting recognition. Using this software without any training is a plus point. Portability to many other languages and tiny size are other plus points. Our software can recognize any letter written normally or near normally. Near hundred percent correct recognition is our goal. It is estimated that the sale of smart phones will see a huge jump in the coming years in India. With this, I think our online handwriting technology will empower many Indians.