Intelligent Training for Tamil Character Recognition using Tesseract OCR

[1]S.Udhayakumar and [2]K.Sibi

[1]Associate Professor, [2]P.G.Student

Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai

Abstract

Tamil character Recognition is one of the challenging tasks in Optical Character Recognition. While the current methodologies employed give higher accuracy for English characters, Tamil characters still face issues. The complexity of Tamil characters when compared to English is multi-fold due to is font style, character combination, etc. These features could be trained using Tesseract which is an open-source optical character recognition (OCR) engine that was developed at Hewlett-Packard, released under Apache2.0 license. Originally developed for English text only, but now efforts have been made to modify the engine and its training system to make them able to deal with other languages and UTF-8 characters. Tesseract needs to know about different shapes of the same character by having different characters separated explicitly. Any language that has different punctuation and numbers is going to be disadvantaged by some of the hard-coded algorithms that assume ASCII punctuation and digits. There have been efforts to “extend” the engine for Tamil. But the accuracy is not comparable with that of English as the style of both languages varies significantly. The scope for ambiguity is high in Tamil (Example - (னை being recognised as னை, வெ being recognised as கிவ). Also Tamil has specific characters like ஃ. Taking these factors into account, the present attempt is to understand the Tesseract OCR and try to find out an efficient/intelligent way to train Tamil characters and give in GPL compatible license so that the community at large benefits from it and enables future work on it .

Introduction

The optical character recognition (OCR) is a process of recognizing printed or handwritten text from scanned or printed document images and converting them into text format. Early versions of OCR need to be programmed with images of each character, and worked on one font at a time. "Intelligent" systems with a high degree of recognition accuracy for most fonts are now common [1]. Tamil character Recognition is one of the challenging tasks in Optical Character Recognition. While the current methodologies employed give higher accuracy for English characters, Tamil characters still face issues. While it’s true that Tamil Character Recognition has its unique feature due to the complex nature of character set when compared to English, the preprocessing can be done with the help of already existing systems .Hence the aim of this paper is to use an existing OCR engine (Tesseract) to train it for recognizing Tamil characters. Tesseract is an open-source optical character recognition (OCR) engine that was developed at Hewlett-Packard between 1984 – 1994, and later in 2005 released under Apache2.0 license [2]. As noted here [3] , the engine performs well with English , but has to be trained for Tamil ,specifically because Indic languages provide greater challenges specifically to classifiers, and also to the other components of OCR systems. Another reason to use Tesseract engine is that its under Apache2.0 license and are compatible with version 3

of the GPL[4] unlike various other OCR's like PonVizhi, i2OCR are either copyrighted or vendor locked or in some cases both [5]. Taking these factors into account, the present attempt is to understand the Tessaract OCR and try to find out an efficient/intelligent way to train Tamil characters and give in GPL compatible license so that the community at large benefits from it and enables future work on it .

Training Architecture

The top-level block diagram of Tesseract is as provided in [3].

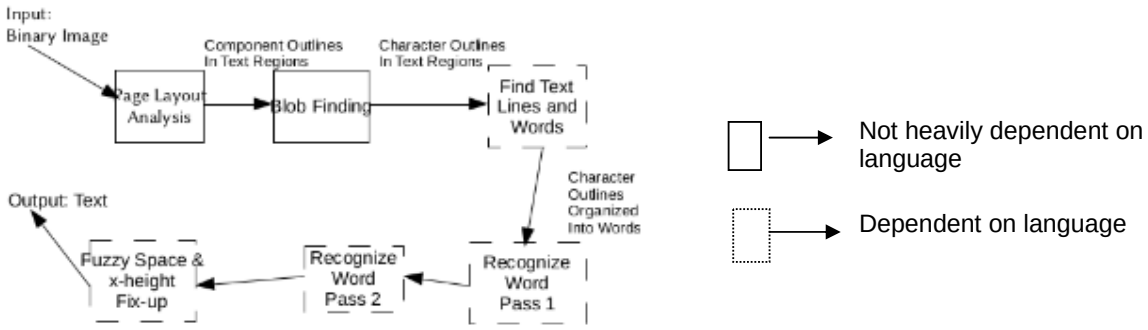


Figure 1: Top-level block diagram

The design architecture can be viewed as 2 parts.

a)Language Independent processes
b)Language Dependent processes

The Language Independent process of Physical Page layout analysis, one of the first steps of OCR, divides an image into areas of text and non- text, as well as splitting multi-column text into columns is given here .[6]

The language dependent modules – Tamil Specific:

Line Finding module:
In the line finding module the author [2] notes that ".....so it is safe to filter out blobs that are smaller than some fraction of the median height, being most likely punctuation, diacritical marks and noise." But Tamil has characters like ∴ which could be misread as punctuations.

Linguistic Analysis
Tesseract contains relatively little linguistic analysis.[2] Hence the Tamil Linguistic analysis is to be incorporated in the engine to find out areas where ambiguity in Grammar rules can be found and how they can be removed .

Training process:
The process used to train Tesseract for recognizing a new language is known as training. The Training data guidelines are provided here [9] with the following to be trained in Tesseract C/C++

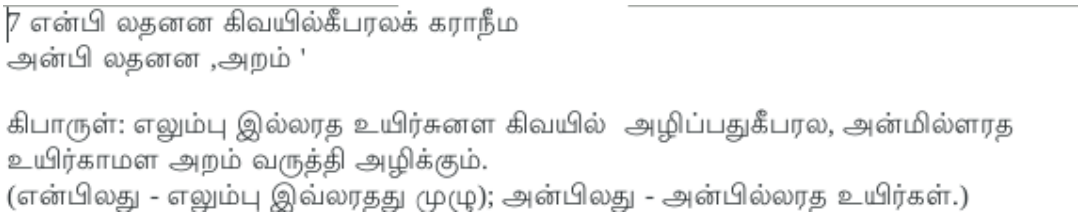
Name	Type	Status	Creator	Description
config	Text	Optional	Manual	Lang-specific engine set tngs if needed
unicharset	Text	Mandatory	unicharset_extractor	The set of recognizable units
unicharambigs	Text	Optional	Manual*	Intrinsic ambiguities for the language
int temp	Binary	Mandatory	mf training	Classifier shape data
pf fntable	Text	Mandatory	mf training	Extra classifier data (num expected features)
normproto	Text	Mandatory	cntraining	Classifier baseline position info
cube-unicharset	Text	Optional	unicharset_extractor	Cube's set of recognizable units
shapetable	Binary	Optional	shapetraining	Indirection between classifier and unicharset
params-model	Text	Optional	Google tool	Alternative method for combining LM & classifie
punc-dawg	Binary	Optional	wordlist2dawg	Pat terns of punctuation around words
word-dawg	Binary	Optional	wordlist2dawg	Main word-list/dictionary language model
number-dawg	Binary	Optional	wordlist2dawg	Acceptable number pat terns (with units?)
freq-dawg	Binary	Optional	wordlist2dawg	Shorter dictionary of frequent words
fixed-length-dawgs	Binary	Deprecated	wordlist2dawg	Was used for CJK
cube-word-dawg	Binary	Optional	wordlist2dawg	Main word-list/dictionary language model for cube
bigram-dawg	Binary	Optional	wordlist2dawg	Word bigram language model
unambig-dawg	Binary	Optional	wordlist2dawg	List of unambiguous words

Intelligent Training process:

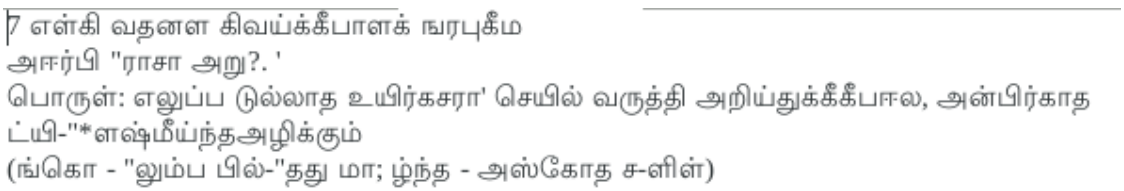
As Tesseract follows a 2 step adaptive classification algorithm, the existing training data set gives different output for same input and also has higher rate of ambiguity. For example, consider the below image

Figure 2 : Standalone input file

when given as a standalone file , the output is given as



but when the same image is a part of larger subset it is recognised as



This is because the training is not proper currently for the engine in Tamil language. As a first

step towards proceeding to any further improvements, the engine has to be trained with various training sets of different fonts. This can be manually done or with the help of jTessBoxEditor v1.0. The jTessBoxEditor has not been incorporated till now for Tamil Language and it's in its initial stages.

Conclusion:

The paper presented a overview of Tamil OCR's present state and extending the Tessaract 3.0 engine for Tamil Language by preparing proper training data set as the first step. While using the current mechanism as described by the training manual, the jTessBoxEditor v1.0 is planned to be used in the subsequent training. The training set is to be ported back to the main project where in it could be further refined to pave way for intelligent training of the Tesseract engine

Acknowledgements:

The authors would like to thanks developers for developing Tesseract engine as well as the ISRI group at UNLV and the Google group for maintaining Tesseract OCR in Apache 2.0 License. The Tesseract -OCR forum and Tesseract -dev forum were used for clearing doubts. The friendly forum's help is acknowledged.

References:

[1] http://en.wikipedia.org/wiki/Optical_character_recognition
[2] Smith, R., "An Overview of the Tesseract OCR Engine" Proc 9th Int. Conf. on Document Analysis and Recognition, 2007, pp629-633.
[3] Adapting the Tesseract open source OCR engine for multilingual OCR ,Ray Smith,Daria Antonova,Dar-Shyang Lee 2009 , Proceedings of the International Workshop on Multilingual OCR
[4] .http://www.apache.org/licenses/LICENSE-2.0
[5] http://ildc.in/tamil/Gist/htm/ocr_spell.htm
[6] Smith, R "Hybrid Page Layout Analysis via Tab-Stop Detection, Document Analysis and Recognition" Proc. 10Th Int. Conf. on Document Analysis and Recognition, 2009.
[7] Layout Analysis,Tesseract Tutorial: DAS 2014 Tours France .
[8] Training Tesseract ,Tesseract Tutorial: DAS 2014 Tours France ,
[9] http://code.google.com/p/tesseract-ocr/wiki/TrainingTesseract3