Support Vector Machine (SVM) Based Classifier For Khmer Printed Character-set Recognition

Mr. SOK Pongsametrey

28th October 2014
Content

- Introduction
- Related Research
- Research Methodology
- Experiment and Discussion
- Conclusion and Future Works
- References
- Demo
- Q & A
INTRODUCTION

- Already some researches on Optical Character Recognition (OCR) on Khmer language

- Some methods are used on different part of the OCR system such as on Segmentation or Recognition but no yet full system from image processing to output currently in use or produce.

- Support Vector Machine (SVM) is a new method for classification and recognition for Khmer OCR but it does not apply for Segmentation
<table>
<thead>
<tr>
<th>Author</th>
<th>Description</th>
<th>Method (Feature/Classifier)</th>
<th>Font/Size</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chey Chanoeurn et al</td>
<td>PCR: 10 Khmer characters: ឃើញការឈុត េស៊ីូរែឬស៊ីូរេជីេស</td>
<td>Lagendre Moment Descriptor Euclidean</td>
<td>1 font</td>
<td>92%</td>
</tr>
<tr>
<td>Chey Chanoeurn et al</td>
<td>PCR: 20 fonts as a template/training and 10 fonts as the testing set with 3 font sizes</td>
<td>Wavelet Descriptors Euclidean</td>
<td>22pt</td>
<td>92.85%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>18pt</td>
<td>91.66%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12pt</td>
<td>89.27%</td>
</tr>
<tr>
<td>Ing Leng leng (PAN)</td>
<td>Khmer Segmentation</td>
<td>Text Band calculation and White space detection</td>
<td>Limon S1, 22pt</td>
<td>92.48%</td>
</tr>
<tr>
<td>Ing Leng leng (PAN)</td>
<td>OCR: Pre-processing: noise removal / Recognition: Discrete Cosine Transform (DCT) based on HMM Toolkit (HTK) / Character mapping / Validation / Sorting / Elimination / Swapping</td>
<td>Discrete Cosine Transform with Hidden Markov Model (HMM)</td>
<td>Limon R1, 22pt</td>
<td>98.88%</td>
</tr>
</tbody>
</table>
## RELATED RESEARCHES

<table>
<thead>
<tr>
<th>Author</th>
<th>Description</th>
<th>Method (Feature/Classifier)</th>
<th>Font/Size</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kruy Vanna</td>
<td>OCR: 3 models proposed from Pre-processing to output</td>
<td>1: Scale Invariant Feature Transform (SIFT + Euclidean)</td>
<td>1./ Khmer OS System</td>
<td>97.90%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: SIFT and Fourier Descriptors, Euclidean (Fuse Distance)</td>
<td>2./ 5 fonts: Khmer OS System, Khmer OS Bokor, Khmer OS Freehand, Khmer OS Muol and Khmer OS MuolPali</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3: Fourier Descriptor, Component’s Hole and Component’s Location + Euclidean (Fuse Distance)</td>
<td>3./ 14 fonts</td>
<td></td>
</tr>
<tr>
<td>Iech Setha</td>
<td>PCR: Segement &amp; Recognition</td>
<td>Edge Detection and Template Matching</td>
<td>Khmer OS Content, 36pt</td>
<td>99%</td>
</tr>
</tbody>
</table>
Problem Statement

• There are some methods already introduced for Khmer OCR in text segmentation and recognition but speed, memory and accuracy is still a concern for a real Khmer OCR system.

• Khmer language has five levels; one main level in the middle, two levels superscript and two levels subscript.
Research Scope

• A new method, Support Vector Machine (SVM) for Khmer OCR via Khmer Printed Character-set in Bitmap format

• One Khmer Font name: Khmer OS Content (or Kh OS Content) for training size of 32pt and target of the document of size: 28pt, 32pt & 36pt

• Apply for any image files

• Segmentation uses Edge Detection from previous research of Ieh Setha et al, 2012
About SVM

• SVM is a binary classifier

• Linear SVM is firstly introduced but data is not always linear

• Non-linear SVM introduces to solve non-linear data but performance is not well performed

• Sequential Minimal Optimization (SMO) is a learning algorithm
• SMO is an optimal algorithm to work with Non-linear SVM
About SVM: Classifier

- Non-linear SVM with following Kernel Trick:
  - Linear SVM Kernel
  - Polynomial SVM Kernel
  - Gaussian SVM Kernel

- Why SVM?
  - In concept, training less dataset both size & font face with acceptable accuracy
  - Minimal consuming time and resource in both training & recognition
METHODOLOGIES

- Two parts have been covered in this section

- **Raw Data Training**
  - Java Implementation

- **SVM**
  - Data Training

- **Edge Detection**
  - Segmentation

- **SVM Classification**
  - Feature Extraction & Recognition

KhmerOCR Application with Accord.NET, C# Implementation
Data Training

• Training the Unicode Character or piece of the character, font size 32pt
• Text rendering is using ASCII (Limon)

Data preparation

• Define Levels of Khmer Characters
• Classified Types of Khmer Characters

Training Module

• A Java application to produce different text file of all charset, main, superscript, subscript character files
• SVM Data Training: Feature Extraction
Data Training: 5 vertical Levels of Cluster

• Middle level characters

• Subscript level characters (another level is lower)

• Superscript level characters (another level is higher)
Data Training – Classified Types

- Connected-point Single Character
  - កប្រង្កឺុ ឫ

- Disconnected-point Single Characters
  - ឈើ ដឹង សុ ឃ
  
- Connected-point Combined-characters
  - កូន សង ដុ ឃ
  
- Disconnected-point Combined-character
  - ស្វែង សម សុ ឃ
  
  \[\text{ស្វែ} = \text{ស្វ} + \text{ុ}, \quad \text{សម} = \text{ស} + \text{ុ}\]
  
  \[\text{ស្វែ} = \text{ស} + \text{ុ}, \quad \text{សម} = \text{ស} + \text{ុ} \Rightarrow \text{ស្វែ} = \text{ស} + \text{ុ} + \sim\]
Data Training – Module

• The process of training is to convert the image character file into the string of binary data (0, 1) of matrix 32x32 dimension

• Small Java Application to convert TrueType font character into image file and generate the matrix
Data Training – SVM

• SVM Data Training
  • There are 500 support vectors
    • All characters set: 250
    • Main level character: 177
    • Superscript character: 21
    • Subscript character: 52
  • Feature extract is an array of 1024 blocks, extracted from binary matrix of 32 x 32
Data Training – SVM

- SVM Data Training
Optical Character Recognition (OCR)

- Segmentation
- Single Character Feature Extraction
- SVM Recognition
- Character Assembling
OCR Process

- Input Image
- Extract Image Feature
- Process Matrix Analysis
- Limon to Unicode Converter
- Output ASCII Character
- Character Assembling
- Process Trained SVM Kernel
- Single Character Feature

Output ASCII Character
OCR - Segmentation

- Segmentation uses Edge Detection in order to separate line and character.
- Line Separation Segmentation is a process to separate line by line of Khmer text in the document
- Character Segmentation is a process to segment the character in a series of character set
OCR – Classification & Recognition

- Uses Multi-class SVM with One-Against-One (OAO) because SVM is binary classifier

- OAO can represent by discarding the redundant option

\[ k = \frac{n(n - 1)}{2} \]
OCR – Character Assembling

- Two issues to solve here

<table>
<thead>
<tr>
<th>Disconnected-point Single/Combined-character</th>
</tr>
</thead>
<tbody>
<tr>
<td>ែ + ែ = មូលែប = មូ</td>
</tr>
<tr>
<td>េ + េ = ធិមុោ = ធិ</td>
</tr>
<tr>
<td>១ + េ = រុបេ = រុ</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SVM Wrong Classification but can solve by character condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ែ + ែ (អូ) = ែអូ = ែ</td>
</tr>
<tr>
<td>េ + េ = េរុ = េ</td>
</tr>
<tr>
<td>ៀ + ៀ = ៀរុ = ៀ</td>
</tr>
<tr>
<td>ៀ + ៀ + ៀ = រុរុប = រុ</td>
</tr>
</tbody>
</table>
## EXPERIMENT

### Experiment Environment

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Microsoft Windows</td>
</tr>
<tr>
<td>Development Tools</td>
<td>Microsoft Visual Studio 2012</td>
</tr>
<tr>
<td>Programming Language</td>
<td>C#</td>
</tr>
<tr>
<td>Running Environment</td>
<td>.Net Framework 3.5</td>
</tr>
<tr>
<td>Framework</td>
<td>Accord.NET 2.9 (latest version 2.11)</td>
</tr>
<tr>
<td>Experiment Inputs</td>
<td>Bitmap files (*.bmp) – Font size: 28pt, 32pt, 36pt</td>
</tr>
<tr>
<td>Algorithms</td>
<td>Edge Detection (Segmentation) SVM (Multi-class, Non-Linear, SMO)</td>
</tr>
<tr>
<td>Experiment Outputs</td>
<td>Text files (training set, log) Machine-code character</td>
</tr>
<tr>
<td>Testing Participation</td>
<td>4 Persons</td>
</tr>
</tbody>
</table>
Experiment Process

• Sample Experiment Data around 3000 characters

• Test on font size: 24pt, 28pt, 32pt, 36pt, 48pt, 60pt

• The testing focus on 3 SVM kernels (Linear, Polynomial & Gaussian) with training algorithms of SMO
Experiment: Example

![Multiclass Support Vector Machines for KhmerOCR Recognition Sample V2](image-url)

<table>
<thead>
<tr>
<th>Character</th>
<th>Classification</th>
<th>Classified Char</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>70</td>
<td>១</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>២</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>៣</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
<td>៤</td>
</tr>
<tr>
<td>5</td>
<td>49</td>
<td>៥</td>
</tr>
</tbody>
</table>
Experiment - Accuracy

Accuracy, font size 36pt

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaussian</td>
<td>98.54%</td>
</tr>
<tr>
<td>Linear</td>
<td>98.44%</td>
</tr>
<tr>
<td>Polynomial</td>
<td>96.78%</td>
</tr>
</tbody>
</table>
Experiment – CPU Usage

- Gaussian: 43%
- Linear: 62%
- Polynomial: 53%
Experiment – Timespent

Gaussian spent 3 sec. 690ms on training and 3sec. 286ms on recognition

- Gaussian
- Linear
- Polynomial

- Training Timespent
  - Gaussian: 3690.47 sec
  - Linear: 1837.00 sec
  - Polynomial: 1030 sec

- Recognition Timespent
  - Gaussian: 30314.42857 sec
  - Linear: 3285.857 sec
  - Polynomial: 3102.286 sec
Evaluation

• Evaluates on Speed, Memory & Accuracy
• According to the experiment, Gaussian Kernel is more better than other kernel so it’s chosen for this evaluation
• Gaussian Kernel, 98.54% accuracy, Avg. CPU of 43% and time spent of more than 3 seconds
Evaluation – Gaussian with other font sizes

• Font size 28pt has 98.17%; 32pt has 98.62%; 36pt has 98.54%; 48pt has 98.61%; 60pt has 99.33%;

• More clear pixels, the more accuracy (segmentation)
Discussion

- SVM with Gaussian Kernel is the best choice for SVM based Khmer OCR.
- Gaussian kernel is a preferred kernel when we don’t know much about the data we are trying to model.
- Some SVM error that can’t solve

<table>
<thead>
<tr>
<th>Found but wrong</th>
<th>Character Expected</th>
<th>Issued on Tested Font Size</th>
<th>Explain</th>
</tr>
</thead>
<tbody>
<tr>
<td>ំ ះ</td>
<td>ំ ះ</td>
<td>28pt, 36pt</td>
<td>Error because of the segmentation, application error</td>
</tr>
<tr>
<td>ំ ះ</td>
<td>ំ ះ</td>
<td>32pt, 36pt</td>
<td>SVM classification error</td>
</tr>
<tr>
<td>ំ ះ</td>
<td>ំ ះ</td>
<td>28pt, 32pt</td>
<td>SVM classification error</td>
</tr>
<tr>
<td>ំ ះ</td>
<td>ំ ះ</td>
<td>32pt</td>
<td>SVM classification error</td>
</tr>
<tr>
<td>ំ ះ</td>
<td>ំ ះ</td>
<td>28pt</td>
<td>SVM classification error</td>
</tr>
<tr>
<td>or ំ ះ</td>
<td>ំ ះ</td>
<td>28pt, 32pt</td>
<td>SVM classification error &amp; Segmentation issue</td>
</tr>
</tbody>
</table>
CONCLUSION

• Khmer characters in bitmap document are well recognized by SVM method
• SVM method uses less CPU (12% better) and time spent on recognition (16 seconds faster)
• The research study with training data of 32pt and test on font size 28pt, 32pt and 36pt got accuracy of 98.17%, 98.62% and 98.54% accordingly
FUTURE WORKS

• Binary data to represent Khmer character seem too limit for classification in SVM, so a study on each Character’s feature to use in SVM will bring more better result
• Apply Multiple Type of Khmer Unicode fonts
• Apply the Scanned document
• Improve Current Segmentation Method
REFERENCES