University of Burgundy
MsCV

Report

Text Detection and Recognition

Majeed Hussain
Akshay Kumar
Abdul Salam Rasmi

05.06.2018

Supervised by
Desire Sidibe

Contents

1 Introduction

Now-a-days with the rapid growth of technology there are many camera based applications are available in portable devices like cell phones, tabs etc. Everyone is able to capture the images easily, but whenever we want to read the text presented in those images are very difficult. This is the main problem in computer vision community.Since so many years, the text detection play very important role in human life it can be helpful in the language translation and navigation. More over text extraction plays a crucial role for blind people when they want to read the text presented in the scene images. By these ways the text recognition and detection can play vital role in humans every day and in future it can be part of so many computer applications

The pipeline of this project goes as follows:
Step 1:
Text Detection.
Step 2:
Text Recognition.

2 Methodology

The Methodology of this is very straight forward first we did detection part which followed by recognition.

2.0.1 Text Detection

For the detection of Text there are lot of approaches in which we followed a strategy which was available in MATHWORKS site in which the steps are followed as follows[Mat]

1. Extracting Maximally Stable Extremal Regions (MSER)

MSER is a method for blob detection in the images; it is a stable connected component of some gray level sets of the image. MSER depends on the threshold of the image, if we give them some threshold value the pixels below that threshold value are âwhiteâ and all those above or equal are black. In our project we choose the minimum threshold value 0.9, MSER detect the objects and all the objects can be filled with different colors in this process some of the regions include the extra background pixels.Those are removed in the canny edge detection process.First of all sweep threshold of intensity from black to white performing a simple luminance thresholding of the image.Then extract the connected components.Later we find a threshold when an extremal region is maximally stable Finally we get the regions descriptors as features of MSER.

2.Removing the Non-Text Regions Based On Basic Geometric Properties

Even though after applying the MSER algorithm it picks out most of the text, it also detects many other stable regions in the image that are not text. You can use a rule-based approach to remove non-text regions. For example, geometric properties of text can be used to filter out non-text regions using simple thresholds.Here we use Use regionprops to measure a few of these properties and remove the non text regions.

3.Removing the Non-Text Regions Based On Stroke Width Variation.

Stroke width is a measure of the width of the curves and lines that make up a character. Text regions tend to have little stroke width variation, whereas non-text regions tend to have larger variations.In the most cases some of the characters have the similar stroke width or thickness so we have to remove the regions where the stroke width exhibits with more variation but most non text regions show a large variation in stroke

width that can be filtered using the coefficient of stroke width variation.The procedure shown above must be applied separately to each detected MSER region.This procedure is follows a for-loop processes in all the regions, and then shows the results of removing the non-text regions using stroke width variation.

4. Merging the Text Regions to get the Final Detected text

In this step for merging the text regions, we compose all the individual characters of detected results.Later we use these results for recognition tasks, such as OCR, or fed to any classifiers to recognize the character the individually.

So for doing this the One approach for merging individual text regions into words or text lines is to first find neighboring text regions and then form a bounding box around these regions. To find neighboring regions, expand the bounding boxes computed earlier with regionprops. This makes the bounding boxes of neighboring text regions overlap such that text regions that are part of the same word or text line form a chain of overlapping bounding boxes.Now we got the expanded boxes of text we use this to recognize the each character from that.

2.0.2 Text Recognition

Character Recognition is a complex problem because of the variety of languages, fonts and styles in which text can be written, and the complex rules of languages etc. Hence, techniques from different disciplines of computer science processing.

Here character dataset used for training is âChar74Kâ dataset. It has 74,000 characters, divided in digits (0-9), capital letters (A-Z) and small letters (a-z),a total of 62 classes are available.So for classifying this characters we used a famous algorithm namely "Support Vector Machine".

Support Vector Machines â SVMs, represent the cutting edge of ranking algorithms and have been receiving special attention from the international scientific community. Many successful applications, based on SVMs, can be found in different domains of knowledge, such as in text categorization, digital image analysis, character recognition and bioinformatics. SVM approach has some advantages compared to others classifiers. They are robust, accurate and very effective even in cases where the number of training samples is small. SVM technique also shows greater ability to generalize and greater likelihood of generating good classifiers.

So rather than giving the each pixel for svm we give some feature for to classify the.The very well known algorithm suited or svm is Histogram of Oriented Gradients algorithm.
Steps to calculate the Histogram of Oriented gradients
1.For an Input image we take the it in the size of 64*128 then it is ready to we are ready

to calculate the HOG descriptor for this image patch.
2.Calculate the Gradient Images:
o calculate a HOG descriptor, we need to first calculate the horizontal and vertical gradients; after all, we want to calculate the histogram of gradients. This is easily achieved by filtering the image with the following kernels.

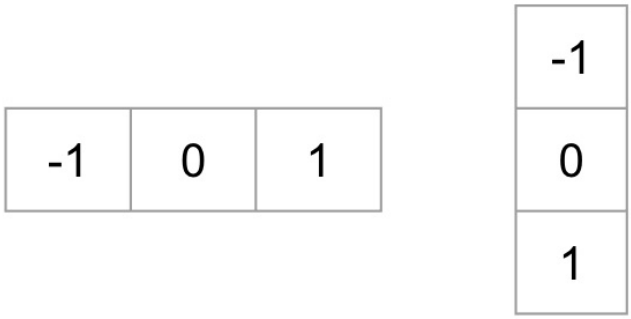


Figure 2.1: Gradient kernel

Next, we can find the magnitude and direction of gradient. The gradient image removed a lot of non-essential information (e.g. constant colored background), but highlighted outlines. In other words, you can look at the gradient image and still easily say there is a person in the picture.At every pixel, the gradient has a magnitude and a direction. For color images, the gradients of the three channels are evaluated (as shown in the figure above). The magnitude of gradient at a pixel is the maximum of the magnitude of gradients of the three channels, and the angle is the angle corresponding to the maximum gradient.

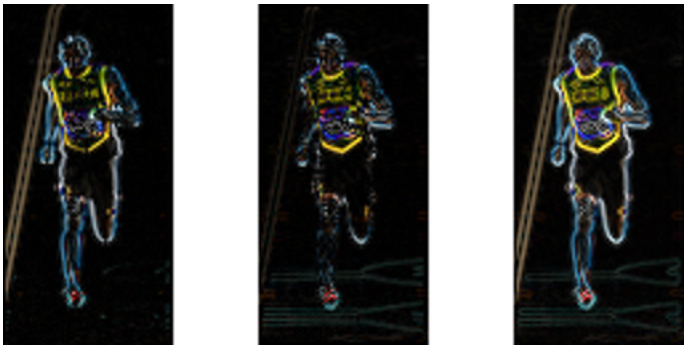
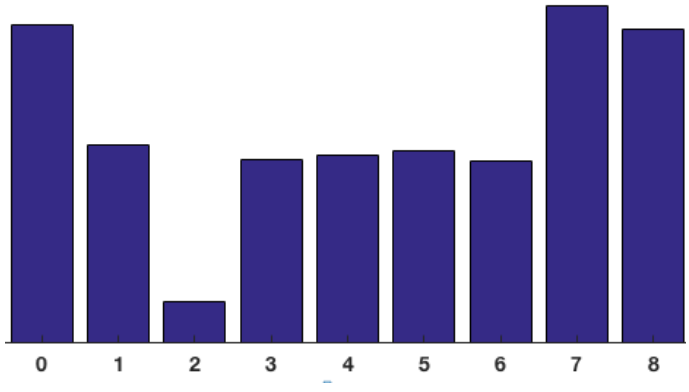


Figure 2.2: Gradients

3.The Next Step is to Calculate Histogram of Gradients in 4Ã4 cells.The histogram is



H]

Figure 2.3: Histogram of Gradients

essentially a vector of 9 bins corresponding to angles 0, 20, 40, 60 â 160.The next step is to create a histogram of gradients in these 8Ã8 cells. The histogram contains 9 bins corresponding to angles 0, 20, 40 â 160.

Later we normalization followed by final Final feature vector.To calculate the final feature vector for the entire image patch, the 16Ã1 vectors are concatenated into one giant vector.After getting this final feature vector we feed this to SVM for classification.Then we have 62 classses now to train.

3 Outputs

1.The following Outputs are obtained in Text Detection and Recognition part.

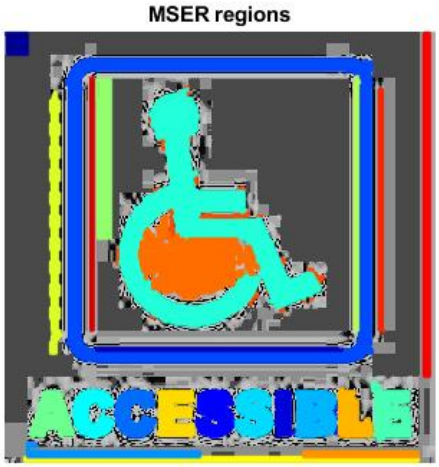


Figure 3.1: MSER regions

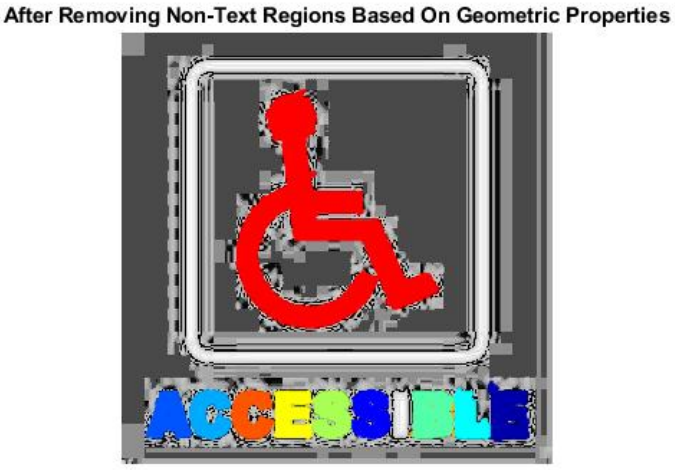


Figure 3.2: After removing non-text regions based on geometric properties

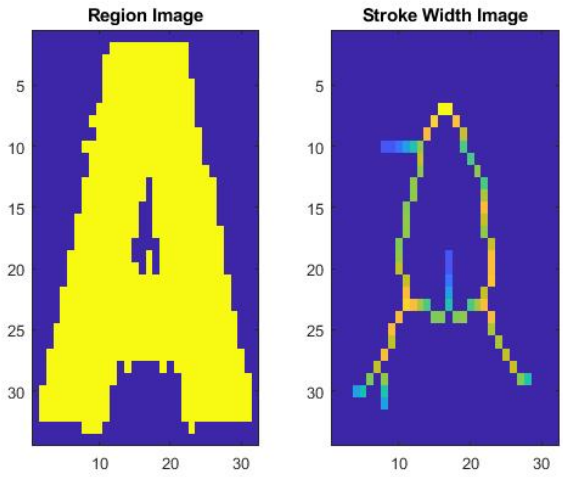


Figure 3.3: Region Image and Stroke width image

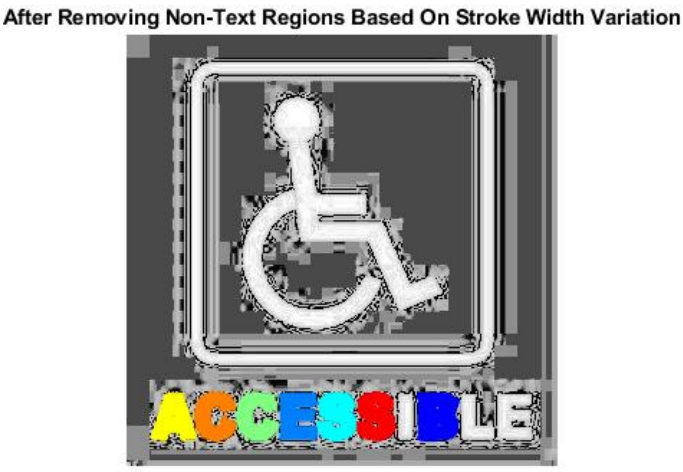


Figure 3.4: After removing non-text regions based on stroke width variation

Figure 3.5: Expanding Bounding boxes

As in our detection part the SWT fails to detect all the characters, it is better to pre-process using canny operator. However it works well in the images with distinct background from the characters.

```
1×7 cell array

  {'A'}    {'C'}    {'C'}    {'E'}    {'S'}    {'B'}    {'B'}
```

Figure 3.6: Final String

4 Conclusion

In conclusion we can say that for the text detection part we can face issues in detecting some natural images because of there different properties one way to get better results is by using enhanced canny edge method which can be efficient than the present method to augment the results .
As far for the Classification as the accuracy was not too high we could train the SVM in such a way that it could get recognize the non text regions too.We can try to increase efficiency in classification by the usage of Convolutional neural networks approach.

Bibliography

[Mat] MATHWORKS: *Text Detection using SWT*.