TEXT DETECTION AND RECOGNITION IN NATURAL IMAGES

Shabana M A[1], Alphy Jose[2], Ani Sunny[3]

[1,2] *Student, Dept. of Computer Science and Engineering, Mar Athanasius College of Engineering, Kerala, India*
[3]*Professor, Dept. of Computer Science and Engineering, Mar Athanasius College of Engineering, Kerala, India*

Abstract - *Text embodied in scene texts are both rich and clear and thus can serve as important cues for a wide range of vision applications, for instance, image understanding, image indexing, video search, geolocation, and automatic navigation. Here we present a unified framework for text detection, recognition in natural images. The proposed system consist of mainly five steps; Candidate generation, Component level classifier, Non text filtering and Text recognition. Among a number of CC (Connected Component) extraction methods, we have adopted the MSER algorithm because it shows good performance with a small computation cost. Filtering of the input image is done using Discrete wavelet transform(DWT) in which edge map of the image is constructed in all directions. The output of the MSER extraction and DWT is combined using logical OR. Filtering by occupation ratio is done by examining the area of the bounding box with respect to the area of the object contained in it. The image is then filtered with the values of stroke widths. Component level classifier will determine adjacency relationship between CCs. Then a classifier that rejects non text blocks among normalized images. This is done by using SVM as classifier. Finally we get the detected text as first output. The detected text is passed to an OCR engine in order to recognize the text.*

***Key Words*: Connected Component (CC), MSER, Discrete Wavelet Transform (DWT), Support Vector Machine (SVM), Stroke Width Transform(SWT), OCR.**

1.INTRODUCTION

With the increasing popularity of practical vision systems and smart phones, text detection in natural scenes becomes a critical yet challenging task. The proposed algorithm can assist numerous applications that require text information extraction from images or videos, such as video search, target geolocation, and automatic navigation. Mainly there are two modules in this project.

- Text Detection
- Text Recognition

In the proposed system consist of mainly four steps. Candidate generation, Component level classifier, Non text filtering and Text recognition. Again candidate generation consist of a number of steps. Among a number of CC (Connected Component) extraction methods, we have adopted the MSER algorithm because it shows good performance with a small computation cost. The MSER(Maximally Stable Extremal Region) algorithm yields CCs that are either darker or brighter than their surroundings. Filtering of the input image is done using Discrete wavelet transform(DWT) in which edge map of the image is constructed in all directions. The output of the MSER extraction and DWT is combined using logical OR. Filtering by occupation ratio is done by examining the area of the bounding box with respect to the area of the object contained in it. Next step is to filter the image with the values of stroke widths.

Then the candidate generation process is completed. We have trained component level classifier that determines adjacency relationship between CCs. In training, we have selected efficient features and trained the classifier with the AdaBoost algorithm. In this way, we are able to detect text. we develop a text or non text classifier that rejects non text blocks among normalized images. This is done by using SVM as classifier. Finally we get the detected text as first output. The detected text is passed to an OCR engine in order to recognize the text.

2. PROPOSED SYSTEM

In the proposed system consist of mainly four steps. Candidate generation, Component level classifier, Non text filtering and Text recognition.

2.1 MSER Extraction

Among a number of CC extraction methods, we have adopted the MSER algorithm because it shows good performance with a small computation cost . This algorithm can be considered as a process to find local binarization results that are stable over a range of thresholds, and this property allows us to find most of the text components. The MSER algorithm yields CCs that are either darker or brighter than their surroundings. We have illustrated brighter CCs by assigning random colors to them. Note that many CCs are overlapping due to the properties of stable regions. More MSER extraction examples which show brighter and darker CCs, respectively.

2.2 Discrete Wavelet Transform

It is a novel method image edge detection using 2-D Discrete Wavelet Transform (DWT) & the results are compared with one of the best classical edge detector: Sobel method for performing edge detection at multiple scales, even in the presence of noise also, that can extract information and perform required processing even at much smaller parts of an image and handle noisy images more efficiently as compared to classical edge detectors.

Wavelet transform is a representation of signals in terms of basic functions that are obtained by dilating and translation a basic wavelet function. We can take a wavelet transform as a tool of low-pass and high-pass filters for edge detection. The wavelet transform has the properties of locality, multi resolution, compression, clustering and persistence. These properties are suitable for most applications in image processing including edge detection.

Classical methods of edge detection involve convolving the image with an operator (a 2-D filter), which is constructed to be sensitive to large gradients in the image while returning values of zero in uniform regions. There are an extremely large number of classical edge detection operators like Roberts, Sobel, Prewitt, Laplacian operators and canny. Operators can be optimized to look for horizontal, vertical or diagonal edges. Classical Edge detectors usually fail to handle images with dull object outline or noisy images, since both the noise and the edges contain high frequency content. Attempts to reduce the noise result in blurred and distorted edges. To reduce the influence of noise, many techniques have been developed. Filtering of images can be done with the Gaussian before edge detection. Methods have also been proposed to approximate the image with a smooth function. Where the classical edge detectors fail to detect edges in noisy images, the proposed edge detection method works efficiently on images corrupted by noise and is able to differentiate between noise and real edges, thus detecting the actual edges.

2.3 Occupation Ratio

It consists of a heuristic rule. This filter runs on the statistical and geometric properties of components, which are very fast to compute. For a connected component c with q foreground pixels, we first compute its bounding box bb(c) (its width and height are denoted by w(c) and h(c), respectively) and the mean as well as standard deviation of the stroke widths, μ(c) and ϭ(c). The definitions of these basic properties and the corresponding valid ranges are summarized below. The components with one or more invalid properties will be taken as non-text regions and discarded. Here we are considering only the occupation ratio as the filtering rule.

Table -1: Heuristic Rules

Property	Definition	Range
width variation	$WV(c) = \frac{\sigma(c)}{\mu(c)}$	[0, 1]
aspect ratio	$AR(c) = \min\{\frac{w(c)}{h(c)}, \frac{h(c)}{w(c)}\}$	[0.1, 1]
occupation ratio	$OR(c) = \frac{q}{w(c) * h(c)}$	[0.1, 1]

2.4 Stroke Width Transform

SWT is a novel image operator that seeks to find the value of stroke width for each image pixel, and demonstrate its use on the task of text detection in natural images. The suggested operator is local and data dependent, which makes it fast and robust enough to eliminate the need for multi-scale computation or scanning windows. Extensive testing shows that the suggested scheme outperforms the latest published algorithms. Its simplicity allows the algorithm to detect texts in many fonts and languages.

The Stroke Width Transform (SWT for short) is a local image operator which computes per pixel the width of the most likely stroke containing the pixel. The output of the SWT is an image of size equal to the size of the input image where each element contains the width of the stroke associated with the pixel. We define a stroke to be a contiguous part of an image that forms a band of a nearly constant width. We do not assume to know the actual width of the stroke but rather recover it.

The initial value of each element of the SWT is set to ∞. In order to recover strokes, we first compute edges in the image using Canny edge detector. After that, a gradient direction d_p of each e edge pixel P is considered. If P lies on a stroke boundary, then d_p must be roughly perpendicular to the orientation of the stroke. We follow the ray r= p+n*d_p, n>0 until another edge pixel q is found. We consider then the gradient direction d_q at pixel q. The SWT operator described

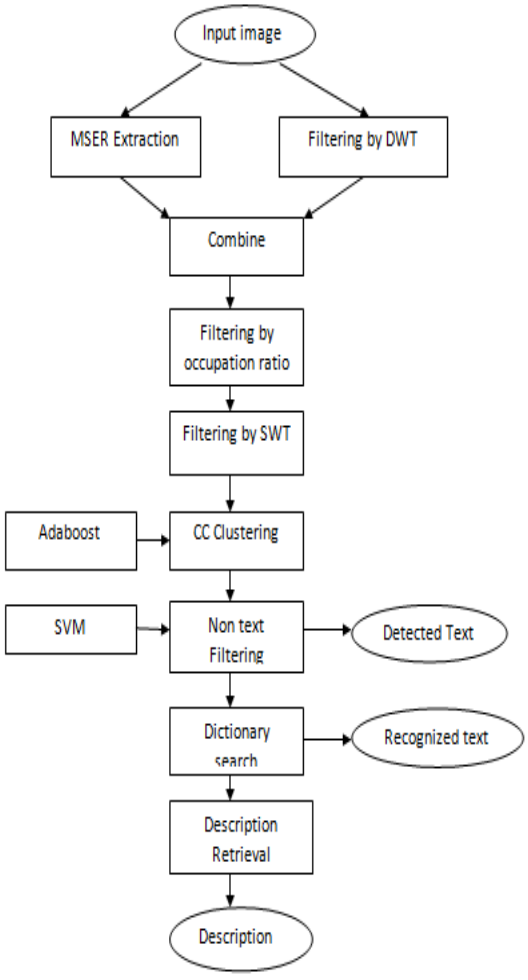

Fig - 1: Overview of proposed system

Here is linear in the number of edge pixels in the image and also linear in the maximal stroke width, determined at the training stage.

To extract connected components from the image, SWT is adopted for its effectiveness and efficiency. In addition, it provides a way to discover connected components from edge map directly. . The next step of this stage is to group the pixels in the SWT image into connected components. The pixels are associated using a simple rule that the ratio of SWT values of neighboring pixels is less than 0.01.

2.5. AdaBoost

Our classifier is based on pair wise relations between CCs, and let us first consider cases that can happen for a CC pair we address a relatively simple problem by adopting an idea in region-based approaches. That is, we adopt a non text filtering block, and we are no longer required to care about other cases. If we have ci ~ cj for some ci , c j ∈ N, it will yield a candidate consisting of non text CCs and this candidate will be rejected at the non text rejection step. Also, we will perform word segmentation as a post processing step. Based on the observations, we build training sets. Specifically, we first obtained sets of CCs by applying the MSER algorithm to a training set. Then, for every pair (ci,cj) ∈ C ×C (i ≠j), we identify its category among 5 cases. A positive set is built by gathering samples corresponding to the case (1) and a negative set by gathering samples corresponding to the case (3) or (4). Samples from other cases were discarded.

$$
\begin{aligned}
&(c_i, c_j) \in \mathcal{C} \times \mathcal{C} \ \ (i \neq j):\\
&\quad 1)\ \ c_i \in \mathcal{T},\ c_j \in \mathcal{T},\ c_i \sim c_j\\
&\quad 2)\ \ c_i \in \mathcal{T},\ c_j \in \mathcal{T},\ c_i \nsim c_j,\ t(c_i) = t(c_j)\\
&\quad 3)\ \ c_i \in \mathcal{T},\ c_j \in \mathcal{T},\ c_i \nsim c_j,\ t(c_i) \neq t(c_j)\\
&\quad 4)\ \ c_i \in \mathcal{T},\ c_j \in \mathcal{N}\\
&\quad 5)\ \ c_i \in \mathcal{N},\ c_j \in \mathcal{N}.
\end{aligned}
$$

2.6. Support Vector Machine

Support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyse data used for classification and regression analysis. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. “Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, each data item is plotted as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, classification is performed by finding the hyper-plane that differentiate the two classes very well.

In order to get final result, we develop a text/nontext classifier that rejects non text blocks among normalized images. In our classification, we do not adopt sophisticated techniques such as cascade structures, since the number of samples to be classified is usually small. However, one challenge for our approach is the variable aspect ratio. One possible approach to this problem is to split the normalized images into patches covering one of the letters and develop a character/non-character classifier. However, character segmentation is not an easy problem and there are examples where this approach may fail. Rather, we split a normalized block into overlapping squares, and develop a classifier that assigns a textness value to each square block. Finally, decision results for all square blocks are integrated so that the original block is classified.

Our feature vector is based on mesh and gradient features. We divide each square into 4 horizontal and vertical ones and extract features.

For a horizontal block Hi (i = 1, 2, 3, 4), we consider

1) The number of white pixels,
2) The number of vertical white-black transitions,

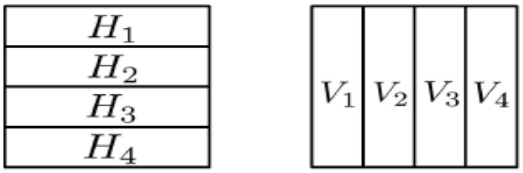

Fig - 2: four horizontal and four vertical blocks.

3) The number of vertical black-white transitions as features, and features for a vertical block is similarly defined.

Here the two classes are text and non text. Initially we train the SVM classifier with a number of images. A number of bounding boxes are created using the region properties of the image. Then we analyse every combination of the bounding box regions. If both are text then 1 is assigned as label otherwise 0 is assigned as label. In the prediction stage these labels for the new image is checked and categorize as text or non-text.

2.7. OCR Engine

Optical character recognition (OCR) method has been used in converting printed text into editable text. OCR is very useful and popular method in various applications. Accuracy of OCR can be dependent on text pre processing and segmentation algorithms. Sometimes it is difficult to retrieve text from the image because of different size, style, orientation, complex background of image etc.

Tesseract is an open source optical character recognition engine. It was modified and improved in 1995 with greater accuracy. It is highly portable. It is more focused towards providing less rejection than accuracy.

Optical character Recognition (OCR) is a conversion of scanned or printed text images, handwritten text into editable text for further processing. This technology allows machine to recognize the text automatically. It is like combination of eye and mind of human body. An eye can view the text from the images but actually the brain processes as well as interprets that extracted text read by eye. In development of computerized OCR system, few problems can occur. First: there is very little visible difference between some letters and digits for computers to understand. For example it might be difficult for the computer to differentiate between digit "0" and letter "o". Second: It might be very difficult to extract text, which is embedded in very dark background or printed on other words or graphics. In 1955, the first commercial system was installed at the reader's digest, which used OCR to input sales report into a computer and then after OCR method has become very helpful in computerizing the physical office documents. There are many applications of OCR, which includes: License plate recognition, image text extraction from natural scene images, extracting text from scanned documents etc.

Today many types of OCR software available in the markets like: Desktop OCR, Server OCR, Web OCR etc. Accuracy rate of any OCR tool varies from 71% to 98%. Many OCR tools are available as of now but only few of them are open source and free. Tesseract is the one of the open source and free OCR software.

An image with the text is given as input to the Tesseract engine that is command based tool. Then it is processed by Tesseract command. Tesseract command takes two arguments: First argument is image file name that contains text and second argument is output text file in which, extracted text is stored. The output file extension is given as .txt by Tesseract, so no need to specify the file extension while specifying the output file name as a second argument in Tesseract command. As Tesseract supports various languages, the language training data file must be kept in the tessdata folder.

After processing is completed, the content of the output file. In simple images with or without colour (gray scale), Tesseract provides results with 100% accuracy. But in the case of some complex images Tesseract provides better accuracy results if the images are in the gray scale mode as compared to colour images.

3. PERFORMANCE ANALYSIS

The experimentation of the proposed algorithm was carried out on a data set consisting of different images such as indoor, outdoor, posters etc. These test images vary with respect to scale, lighting and orientation of text in the image. Currently the data set consists of 10 images.

3.1 Performance of Text Detection

Text detection is done on various images in which the number of object in the image varies. The results show that as the number of objects in the image varies the accuracy get reduced since the image is becoming complex.

Based on such an analysis with different types of images with varying complexity we have drawn a graph. In which the x-axis represents the images and y-axis represents the accuracy. As we move from left to right in x-axis the complexity of images is increasing.

Fig - 3: Image 1 Stages

The system is tested with various images having different orientations. Some examples of such images in different stages is shown bellow. From the analysis it is clear that the system is efficient to detect text having different orientations up to a large extent from the images. The analysis shows that the system fails to images in which text is making a large angle with the horizontal.

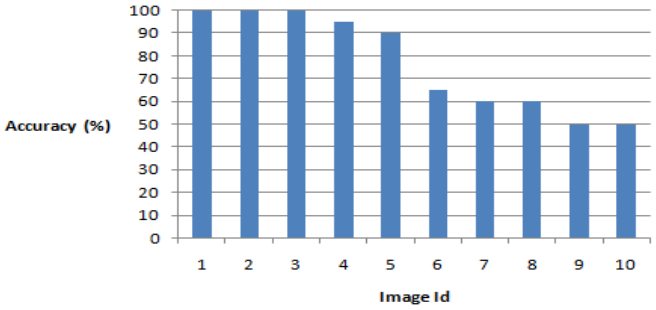

Chart - 1: Performance Analysis 1

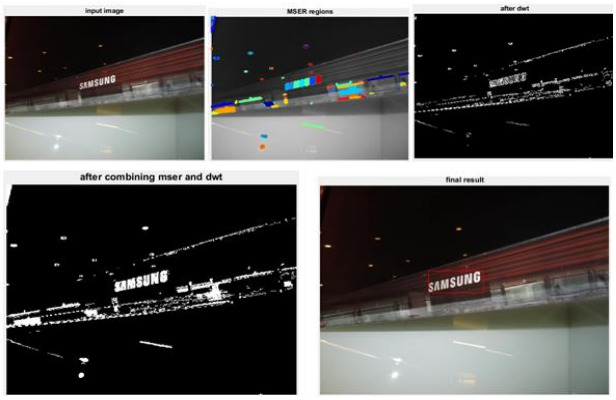

Fig - 4: Image 2 Stages

Based on such an analysis with different types of images with varying orientation we have drawn a graph. In which the x-axis represents the images and y-axis represents the accuracy. As we move from left to right in x-axis the Orientation of images with respect to horizontal line is increasing.

3.2 Performance of Text Recognition

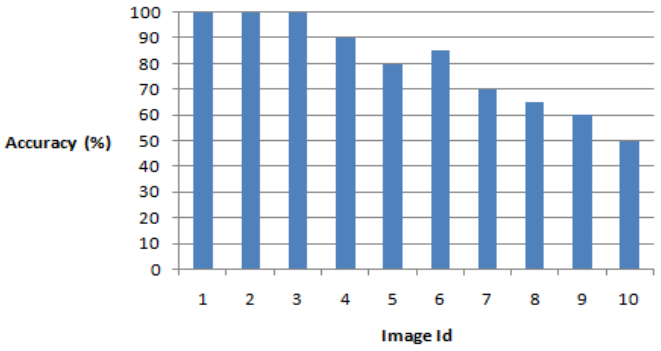

Chart - 2: Performance Analysis 2

Text recognition is done on various images in which the number of object in the image varies. The results show that as the number of objects in the image, orientation and the connection between the characters varies the accuracy get affected since the image is becoming complex to recognize each character.

3. CONCLUSIONS

High level semantics embodied in scene texts are both rich and clear and thus can serve as important cues for a wide

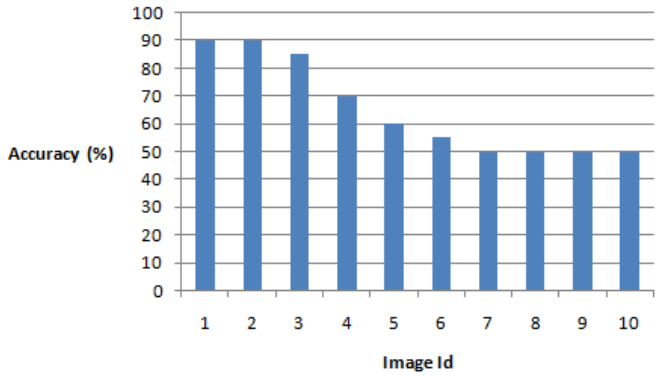

Chart - 3: Performance Analysis 3

range of vision applications, for instance, image understanding, image indexing, video search, geolocation, and automatic navigation. Here we implemented a unified framework for text detection, recognition and retrieval of details about the recognized text in natural images. Mainly there are two modules in this project. Text detection and recognition. The text detection is done with MSER regions, DWT, SWT etc. Also a clustering and a classification is also done. The text recognition is done with OCR engine. The results shows that as the complexity, orientation, and connection between two characters varies the possibility for detection and recognition also varies.

REFERENCES

[1] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in Proc. IEEE CVPR, Jun./Jul. 2004, pp. II-366–II-373.

[2] D. Chen, J.-M. Odobez, and H. Bourlard, "Text detection and recognition in images and video frames," Pattern Recognit., vol. 37, no. 3, pp. 595–608, 2004.

[3] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in Proc. 10th ACCV, 2010, pp. 770–783.

[4] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in Proc. IEEE ICCV, Nov. 2011, pp. 1457–1464.

[5] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in Proc. IEEE CVPR, Jun. 2010, pp. 2963–2970.

[6] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in Proc. IEEE CVPR, Jun. 2012, pp. 1083–1090.

[7] Cong Yao, Xiang Bai, "A Unified Framework for Multioriented Text Detection and Recognition", IEEE Transactions On Image Processing, Vol. 23, No. 11, November 2014.

[8] Hyung Il Koo, "Scene Text Detection via Connected Component Clustering and Nontext Filtering", IEEE Transactions On Image Processing, Vol. 22, No. 6, June 2013.