Voice Synthesis of Text in Images using Supervised Learning

Cephas Paul Edward V
Student, Department of Computer Science and Engineering, Anna University, Tiruchirappalli, India.

Abstract— This paper deals with the synthesis of voice/speech for the text obtained from the image after detection and recognition using supervised learning. The main advantage over the traditional OCR systems is that, it can potentially recognize more complicated scene images with numerous variations in font, texture and background. In this proposed system, we extract various features from scene images after pre-processing and predict the actual text using support vector machines (SVM), unlike certain existing approaches employ a variant of k-means clustering for this. The result of the recognition is synthesized into voice/speech using suitable text-to-speech library. This system can be used to aid the illiterate and people with poor eye sight.

Index Terms— K- Means Clustering , Optical Character Recognition, SVM , Supervised Learning, Support Vector Machine, Text extraction, Voice synthesis .

1 INTRODUCTION

Extracting text from images such as name boards, flux boards is not as easy to detect and extract text as in Optical Character Recognition (OCR). Optical Character Recognition involves identifying individual characters of text on a plain, monochrome background. But in the case of images, the text is surrounded by a complex background such as multi-colored backgrounds and complicated objects, which involves algorithms that are more complex than plain OCR to extract the text from images without considering the background. Adding still more complexity the text in the images also do not confine themselves to a standardized font under certain circumstances. Thus, plain OCR becomes helpless in this situation. For the recognition process, SVM (Support Vector Machines) are used. SVM was initially designed for binary classification, which involves only two classes. It can be extended to solve a multi-class classification problem such as character recognition from images.

2 EXISTING APPROACHES

Text data present in images and video contain useful information for automatic annotation, indexing, and structuring of images. Extraction of this information involves detection, localization, tracking, extraction, enhancement, and recognition of the text from a given image. However, variations of text due to differences in size, style, orientation, and alignment, as well as low image contrast and complex background make the problem of automatic text extraction extremely challenging.

There is no prior information on whether or not the input image contains any text, the existence or non-existence of text in the image must be determined. Several approaches like Region-based and Stroke-Width transform based approaches assume that certain types of video frame or image contain text. This is a common assumption for scanned images.

2.1 REGION-BASED METHODS

Region-based methods use the properties of the color or gray-scale in a text region or their differences with the corresponding properties of the background. These methods can be further divided into two sub-approaches: connected component (CC)-based and edge-based. These two approaches work in a bottom-up fashion; by identifying sub-structures, such as CCs or edges, and then merging these sub-structures to mark bounding boxes for text. Note that some approaches use a combination of both CC-based and edge-based methods.

2.2 CC-BASED METHODS

CC-based methods use a bottom-up approach by grouping small components into successively larger components until all regions are identified in the image. A geometrical analysis is needed to merge the text components using the spatial arrangement of the components so as to filter out non-text components and mark the boundaries of the text regions.

Ohya et al. presented a four-stage method: (i) binarization based on local thresholding, (ii) tentative character component detection using gray-level difference, (iii) character recognition for calculating the similarities between the character candidates and the standard patterns stored in a database, and (iv) relaxation operation to update the similarities. They were able to extract and recognize characters, including multi-segment characters, under varying illuminating conditions, sizes, positions, and fonts when dealing with scene text images, such as freight train, signboard, etc. However, binary segmentation is inappropriate for video documents, which can have several objects with different gray levels and high levels of noise and variations in illumination. Furthermore, this approach places several restrictions related to text alignment, such as upright and not connected, as well as the color of the text (monochrome). Based on experiments involving 100 images, their recall rate of text localization was 85.4% and the character recognition rate was 66.5%. Kim segments an image using color clustering in a color histogram in the RGB space. Non-text components, such as long horizontal lines and image boundaries, are eliminated. Then, horizontal text lines and text segments are extracted based on an iterative projection profile analysis. In the post-processing stage, these text segments are merged based on heuristics. Since several threshold values need to be determined empirically, this approach is not suitable as a general-purpose text localizer. Experiments were performed with 50 video images, including various character sizes and styles, and a localization rate of 87% was reported.

2.3 CONNECTED COMPONENT CLUSTERING AND NON-TEXT FILTERING

This approach is based on two machine learning algorithms namely Maximum stable extremal region algorithm and AdaBoost Classification algorithm. The first one allows to generate candidate word regions and the other filters out nontext ones. To be precise, connected components (CCs) are extracted in images by using the maximally stable extremal region algorithm. These extracted CCs are partitioned into clusters so that candidate regions can be generated. Unlike conventional methods relying on heuristic rules in clustering, an AdaBoost classifier is trained that determines the adjacency relationship and cluster CCs by using their pair wise relations. Then, the candidate word regions are normalized and determined whether each region contains text or not. Since the scale, skew, and color of each candidate can be estimated from CCs, a text/nontext classifier for normalized images is developed. This classifier is based on multilayer perceptron and recall and precision rates can be controlled with a single free parameter.s

2.4 EDGE BASED DETECTION

The video indexing system introduced by Sato et al. combines closed caption extraction with superimposed caption (artificial text) extraction. The text extraction algorithm is based on the fact that text consists of strokes with high contrast. It searches for vertical edges which are grouped into rectangles. The authors recognized the necessity to improve the quality of the text before passing an OCR step. Consequently, they perform an interpolation of the detected text rectangles before integrating multiple frames into a single enhanced image by taking the minimum/maximum value for each pixel. They also introduced an OCR step based on a correlation measure. A similar method using edge detection and edge clustering has been proposed by Agnihotri and Dimitrova . Wu, Manmatha and Riseman combine the search for vertical edges with a texture filter to detect text. Unfortunately, these binary edgeclustering techniques are sensitive to the binarization step of the edge detectors. A similar approach has been developed by Myers et al. However, the authors concentrate on the correction of perspective distortions of scene text after the detection. Therefore, the text must be large so that baselines andvanishing points can be found. Since the detection of these features is not always possible, assumptions on the imaging geometry need to be made.

2.5 DETECTION THROUGH SEGMENTATION AND SPATIAL GROUPING

The first text detection algorithms, introduced by the document processing community for the extraction of text from colored journal images and web pages, segment characters before grouping them to words and lines. Jain et al. perform a color space reduction followed by color segmentation and spatial regrouping to detect text. Although processing of touching characters is considered by the authors, the segmentation phase presents major problems in the case of low quality documents, especially video sequences. A similar approach, which gives impressive results on text with large fonts, has been presented by Lienhart. A segmentation algorithm and regrouping algorithm are combined with a filter detecting high local contrast, which results in a method which is more adapted to text of low quality. Still, the author cannot demonstrate the reliability of his algorithm in the case of small text. False alarms are removed by texture analysis, and tracking is performed on character level, which might pose considerable problems in the case of text as presented in figure 1b. Similar methods working on color clustering or thresholding followed by a regrouping of components have been presented by Lee and Kankanhalli, by Zhou and Lopresti and by Sobottka et al. Hase et al. cluster the components and allow for spatial arrangements which follow a quadratic function.

2.6 TEXTURE BASED METHODS

Texture-based methods use the observation that text in images have distinct textural properties that distinguish them from the background. The techniques based on Gabor filters, Wavelet, FFT, spatial variance, etc. can be used to detect the textural properties of a text region in an image.Zhong et al. use the local spatial variations in a gray-scale image to locate text regions with a high variance. They utilize a horizontal window of size 1×21 to compute the spatial variance for pixels in a local neighborhood. Then the horizontal edges in an image are identified using a Canny edge detector, and the small edge components are merged into longer lines. From this edge image, edges with opposite directions are paired into the lower and upper boundaries of a text line. However, this approach can only detect horizontal components with a large variation compared to the background.

2.7 SLIDING WINDOW BASED APPROACH

In this type of approach, the color RGB image is binarized using a particular threshold value. A sliding window, initially of size 2x2 is slided over the binarized image, scanning its entire area. The window size is gradually increased until it is finally the size of the input image. A binary SVM is trained to detect whether each of the sliding windows contain text character or not. Later all text character containing sub-images are fed to a variant of k-means clustering algorithm to cluster the various characters for feature learning. There are totally 62 classes, 10 digits, 26 upper-case and 26 lower-case letters.

3 SYSTEM ARCHITECTURE OVERVIEW

In the proposed system, synthesis of voice/speech for the text is obtained from image after detection and recognition using supervised learning. Before the system is ready to predict input images, it should be trained with numerous samples. For easy pre-processing and training process, automatically all the training images should be read one by one and processed. For this purpose the File Scanner module is used. It reads all image files within the specified directory into a pixel array suitable for further processing. The color image, as such, is difficult for processing. So, it is preprocessed using certain filters to alter poor brightness and contrast conditions. Then, it is converted into a binary image. A binary image has only two possible pixel values – Black or white (0 or 1). A binary image so obtained can be of two types based on the image characteristics.

- background has black pixels and object has white pixels
- background has white pixels and object has black pixels

It would be better if the background is standardized as white and objects with black pixels. Instead of the sliding window fashion[1], a simpler, less computationally intensive method is used.

The connected regions are identified. The binary image is scanned from zeroth pixel position, which is at the top- left corner of the image. If the pixel encountered is an object pixel, it checked whether its neighbors are allotted with labels. If yes, the neighboring pixel values is assigned to the current pixel. Or else, a new label is assigned. This process is continued until all pixels are labeled. A second pass is made over the image, equalizing all assigned labels.

The label matrix so obtained is scanned over again and for each unique label, the pixel regions are extracted into separate image files. The label values are replaced with values of BLACK. The images so obtained have so much of extra white space and are found at random locations.

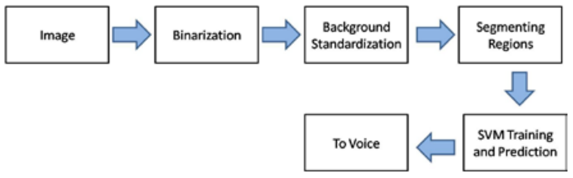

Fig. 1. System Architecture

These images have to be trimmed off their white space and reduced to a standard size, Standard size chosen is 32 x 32, suitable for further processing. The standardized 32 x 32 images are further sent for Feature Extraction. The features are stored onto a file – features.dat, to be fed as input to the Support Vector Machine Algorithm. Then, training started and the Features are extracted and written to features.dat alongwith their class labels.

Training and Testing are completed. Now, the system is capable of predicting any other image given as input. Now, the predicted text is converted to voice using suitable Text-to-speech libraries. Thus the synthesis of voice/speech from text in images is completed successfully. Further, improvements of the system can be incorporated and the system can be still extended like integrating the software into a J2ME mobile such that anyone can carry it around and make use of the system.

4 DETAILED DESCRIPTION

4.1 FILE SCANNER

The project involves the characters segmented into individual images. These are to be read one by one and trimmed and stored as 32 x 32 images separately. Again, similarly one by one the pixel values are extracted and processed. For this purpose, File Scanner module is implemented. It reads all files in a given directory location and facilitates further processing. Thus it automates the file reading process for any number of files.

4.2 READING THE IMAGE

The values of the different pixels of the image are read into an array making use of Image IO functions in Java. A BufferedImage object is declared. This acts as a buffer between the image stored on disk and the numeric array where the pixel values are to be stored. The image is read using the read method of the ImageIO class into the BufferedImage object. An array of the size of the image is declared and the numeric pixel values are stored into it using the getRGB() method.

4.3 PIXEL VALUES AND THE RGBA COLOR MODEL

The color model in practice is the RGBA color model where R, G, B and A stand for Red, Green, Blue and Alpha components respectively. While it is sometimes described as a separate color model, actually it is the usage of RGB color model with extra information. Each of these component requires 8 bits and hence totally a pixel requires 32 bits. Alpha component represents the opacity of the image. Modifying the alpha value, modifies the transparency of the image. RGBA values are represented as a sequence of 8 hexadecimal digits. Each pair of hexadecimal digits represents the values of Alpha, Red, Blue or Green channel. Hence to get these A, R and G components we have to shift the pixel value by 24, 16 and 8 bits. B does not require any shifting. PNG (Portable Network Graphics) format is an image format that uses the RGBA color model.The PNG format is a raster graphics file format that supports lossless data compression.

4.4 BINARIZATION

Color images are not suitable for processing and feature extraction. They are filtered and preprocessed then it would be better if it is binarized. A binary image consists of only two possible values for any pixel. Black or white (0 or 1). The images are binarized using BufferedImage and setting its property to BufferedImage.TYPE_BYTE_BINARY. Virtually, we create a new Graphics canvas and draw the image using drawImage(). Finally the converted binary image is written to the disk storage using the write method of the ImageIO class.

4.5 LABELLING REGIONS

We just identify the connected regions in the binary image. The binary image is scanned from zeroth pixel position which is located at the top-left corner.

If the pixel encountered is an object pixel, it checked whether its neighbors are allotted with labels.

- If yes, the neighboring pixel values is assigned to the current pixel.
- Or else, a new label is assigned.

This process is continued until all pixels are labeled. A second pass is made over the labels to replace each label with its equivalence class. Thus we obtain an array of labels of various regions. To preserve the order of characters in text while labeling, it was discovered that the image has to be rotated by 90 degrees. To do this, AffineTranform class and its methods are used. After rotating the image by 90 degree segmentation process is performed.

In geometry, an affine transformation or an affine map or an affinity is a function between affine spaces which preserves points, straight lines and planes. Also, set of parallel lines remain parallel after an affine transformation. An affine transformation does not necessarily preserve angles between lines or distances between points, though it does preserve ratios of distances between points lying on a straight line. Examples of affine transformations include translation, scaling, similarity transformation, reflection and rotation.

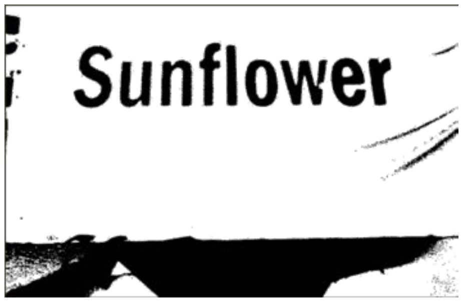

Fig. 2. Binarized Image

4.6 SEGMENTING THE REGIONS

The process of labeling paves way for easy segmentation. The label matrix so obtained is scanned over again and for each unique label, the pixel regions are extracted into separate image files.

4.7 RESIZING THE IMAGES

The images are converted into a standard size of 32 x 32. This is accomplished using the setRenderingHint method with parameter RenderingHints.KEY_INTERPOLATION.

4.8 FEATURE EXTRACTION

The raw pixel values of the 32 X 32 images are used as features. The features are stored in the file named features.dat, which is used to train the SVM. The features are numbered from 0 to 1023 and written one after another beginning with their class labels. A space is included between the class label and the index 0 of the first pixel. A newline character is introduced only after the 1024th pixel is written.

4.9 PREDICTION

Once the SVM is trained and the model file is generated, it can be used to predict any text characters. The predicted labels are written to a file.

4.10 SPEECH SYNTHESIS

The labels from the file are extracted. Decimal points and fractional part is removed and converted to suitable characters. FreeTTS, a java library for speech synthesis can be used.It requires declaration of objects of the classes Voice and VoiceManager. The voice used is Kevin16 which is a 16Khz voice. Finally, the word is spoken using the speak method.

5 MULTICLASS SVM

In machine learning, Support Vector Machines (SVMs, also support vector networks) are supervised learning models are used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. 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. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

There are many linear classifiers (hyper planes) that separate the data. However only one of these achieves maximum separation. The reason we need it is because if we use a hyper plane to classify, it might end up closer to one set of datasets compared to others and we do not want this to happen and thus we see that the concept of maximum margin classifier or hyper plane as an apparent solution.

The aim of SVM algorithm is to find an optical hyper plane which separates the data efficiently. The vectors H1 and H2 are called support vectors. They lie on the boundary of each class and parallel to the Optical Hyper plane, H. This optical hyper plane H, maximizes the margin value.

Let D be a dataset consisting of 'n' points. (xi,yi) is any point in the dataset. Let yi = 1 and yi=-1 indicate the class to which xi belongs. The aim is to find the maximum-margin hyperplane that divides the points having yi=1 and those having yi=-1. Any hyperplane can be represented as

$$w.x - b = 0$$

where . is the dot product of two vectors and w is the normal to the hyperplane.The term ‖w‖ is the offset of the hyperplane from origin along w.

Hence the two hyperplanes corresponding to yi = 1 and yi=-1 are

$$w.x – b = 1$$

$$w.x – b = -1$$

By geometry, is to minimize 2/ ‖ w ‖ . is the distance between the two hyperplanes, so our aim is to minimize ‖ w ‖

SVM was initially designed for binary classification, which involves only two classes. It can be extended to solve a multi-class classification problem such as character recognition from images.

REFERENCES

[1] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J. Wu, Andrew Y. Ng (2011) in International Conference on document analysis and recognition ICDAR, Beijing, China.

[2] Christian Wolf and Jean-Michel Jolion (2004) "Model based text detection in images and videos: a learning approach" LIRIS INSA de Lyon Bˆat. Jules Verne 20, Avenue Albert Einstein Villeurbanne, 69621 cedex, France.

[3] Chucai Yi, Student Member, IEEE, and YingliTian, (2012) "Localizing Text in Scene Images by Boundary Clustering, Stroke Segmentation, and String Fragment Classification" IEEE Transactions on Image Processing, Vol.21, No. 9.

[4] Gee-Sern Hsu, Member IEEE, Jiun-Chang Chen, and Yu-Zu Chung (2013)"Application-Oriented License Plate Recognition" IEEE Transactions on Vehicular Technology, Vol.62, No. 2.

[5] Huizhong Chen, Sam S. Tsai, Georg Schroth, David M. Chen, RadekGrzeszczuk and Bernd Girod (2011) "Robust Text Detection in Natural Images With Edge-Enhanced maximally Stable Extreme Regions"

[6] Hyung Il Koo, Member IEEE and Duck Hoon Kim (2013) "Scene Text Detection via Connected Component Clustering and Nontext Filtering", IEEE Transactions on Image Processing, Vol.22, No. 6.

[7] JakaSodnik and SaSoTomaZIE (2009) "Spatial Speaker: 3D Java Text-to-Speech Converter" World Congress on Engineering and Computer Science 2009 Vol II WCECS 2009, October 20-22, 2009, San Francisco, USA.

[8] Keechul Jung, Kwang In Kim, Anil K. Jain (2010) "Text Information Extraction in Images and Video: A Survey" [Online] Available: search.proquest.com

[9] Palaiahnakote Shivakumara, Rushi Padhuman Screenhar, TrungQuyPhan, Shijian Lu and Chew Lim Tan (2012) "Multioriented Video Scene Text Detection through Bayesian Classification and Boundary Growing" IEEE Transactions on Circuits and Systems for Video technology, Vol.22, No. 8.