Segmentation and Recognition of Gujarati Printed Numerals from Image

Jagin M. Patel
M. K. Institute of Computer Studies, Bharuch,
Gujarat, India.

Apurva A. Desai
Dept. of Computer Science, Veer Narmad South Gujarat
University, Surat, Gujarat, India

Abstract-- Recognition of numerals in an image is very difficult issue because there is no prior knowledge about the color of numerals, distribution of lighting, noise, complexity of background, shape similarity etc. To the best of our knowledge, there is no work reported for segmentation and recognition of Gujarati printed numerals from an image. In this paper, we have discussed the problems of segmentation and recognition of Gujarati printed numerals from an image. We have also adopted simple approach to address this problem. We have used edge detection, dilation and connected component analysis for segmentation. Various heuristics are used to find candidate object for Gujarati numeral. For classification, template matching is used. The experimental results have proved its effectiveness. We have tested our model on different types of images and have achieved success of more than 95%.

I. INTRODUCTION

In an Image processing research, it is necessary to derive better tools that allow us to understand image content, meaning, and importance. Accessibility of information from image is important for various reasons. It is necessary to recognize the text in an image on web pages, for effective indexing and for presentation by non-visual means [1].

There are mainly two objectives to develop the model. First, is to help children in learning and recognizing Gujarati numerals in an interactive and playful manner through computers and second, is to help Content based image retrieval. Images like – Charts, Logos, Graphs, Maps, Headers, Footers and Equations – may contain numeral and character. It is possible that these types of images have important information. Current tools and search engines can find out any numeral, character and text written in text document but are unable to extract any numeral, character and text embedded or found inside the images. Therefore, important information is not accessible using current tools.

A. Gujarati Language

Gujarati is an Indo-Aryan language, one of the official languages of India and native language to the Indian state of Gujarat. It is written from left to right and was evolved from the Devanagari script. It has its own character set. It has 12 vowels (known as Swar), 34 consonants (known as Vyanjan) and 10 digits (known as numerals). The Gujarati numerals are shown in table 1.

Table 1. Gujarati numeral set

English Digit	Gujarati Numeral	Gujarati Name	Pronunciation
0	૦	શૂન્ય	Shunya
1	૧	એક	Ek
2	૨	બે	Be
3	૩	ત્રણ	Tran
4	૪	ચાર	Chaar
5	૫	પાંચ	Panch
6	૬	છ	Chha
7	૭	સાત	Saat
8	૮	આઠ	Aath
9	૯	નવ	Nav

B. Challenges in Gujarati Numeral Recognition

Gujarati numerals are very peculiar in nature. The numerals in Gujarati language are based on sharp curves and hardly any straight line is available in it. Shapes of few Gujarati numerals are more or less similar, so these numerals are confusing and often misunderstood [2,3].

Let us focus on the various situations:

i. The shape similarities occur between some numerals and other Gujarati characters. A little change or noise in character can create confusion. For example following numerals and characters are similar in appearance:
   a. numeral ૦ , characters degree sign (^{0}) and modifier (Anuswar)
   b. numeral ૧ and character વ (Va)
   c. numeral ૪ and character જ (Ja)
   d. numeral ૬, character ક (Ka) and ફ (Ph)

ii. In Gujarati character set, shape of following numerals and characters are exactly same:
   a. Numeral ૨ and character ર (Ra)
   b. Numeral ૫ and character પ (Pa)

c. Numeral ૯ and character ૯ (Half ધ)

iii. In most of the languages, numerals are made of connected strokes and/or curve. In Gujarati language, one numeral ૯ is made of disconnected curve and stroke: "(" and "- ".

iv. The shape of three numerals ૧, ૪ and ૭ includes the shape of another numeral ૦ (zero).

v. Few numerals have width more than their height[3].

Because of all these situations, Gujarati numeral recognition becomes more difficult and needs extra treatment as compared to other languages. We cannot use character or numeral segmentation and recognition methods which are readily available for other languages.

This paper is divided into six different sections. This is an introductory section. Section two is Related Work. Section three describes template creation process. Section four describes the proposed model. Fifth section gives experimental result, and last section gives conclusion and path ahead.

II. Related Work

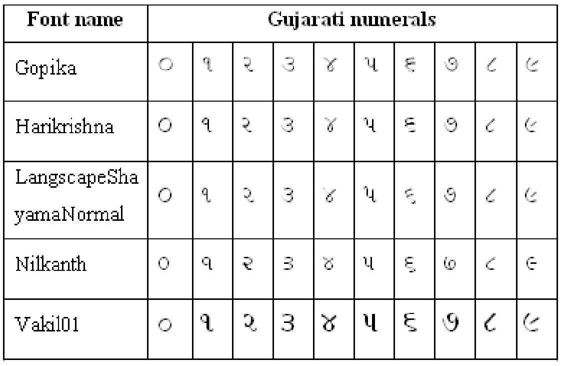
In literature, a number of methods have been proposed for the segmentation or extraction of characters (printed as well as handwritten), identification of text region and localization of text in image. Work is found for international languages like English, Chinese etc. However, to the best of our knowledge, there is no work reported for segmentation and recognition of Gujarati printed numerals from image.

In literature, according to Desai [3], perhaps the first ever work of character recognition for Gujarati printed language was presented by Antani and Agnihotri [4]. They used Euclidean minimum distance classifier (EMDC), hamming distance classifier (HDC) and K- Nearest Neighbor to classify various Gujarati printed characters. This method describes the classification of a subset of printed or digitized Gujarati characters. This method was tested on a small data set. All characters in the sample set are of same size. It has low recognition rate of 67 %. Keunhwi et al. [5] presented character segmentation and recognition algorithm of text region in steel images. In this algorithm, the authors combined profile analysis and recognition based method to separate the touching character. In the preprocessing step, gray-scale images are converted to binary images using the method of adjusting brightness and contrast. This algorithm separates various text region images and then characters are classified using one-to one SVM. Success rate of this algorithm is 94.02%. Grover et al. [6] proposed a simple edge based algorithm to extract text from document images. In this method, first, edge detection in gray-scale image is done using 3 x 3 Sobel operator, non-maximum suppression and thresholding. Then edge image is divided into small non-overlapping blocks. Edge-based feature for each block is calculated and used to classify whether block contains text or not. This method is independent of the orientation of the text, background of the image as well as font size. In year 2010, Desai [2] described how various parameters like paper, writing style, pen etc. affect handwritten Gujarati numeral recognition. He also presented an OCR system for handwritten Gujarati numbers. In preprocessing steps all numerals are put in the standard form. To do so, smoothing, thinning, skew detection, correction and normalization is performed. First, contrast is adjusted by contrast limited adaptive histogram equalization (CLAHE) algorithm. The boundaries are then smoothed out by median filter of 3x3 neighborhoods. The image of numeral is then normalized to the size of 16x16 pixels using nearest neighbor interpolation. He used vectors of four profiles (horizontal, vertical and two diagonal) as an abstract feature for identification of a numeral. To deal with the skew problem, five more patterns for each numeral are created in both clockwise and anticlockwise directions with the difference of 2 degrees each up to 10^o. A multi-layered feed forward back propagation neural network is used for Gujarati numeral classification. This system achieved success rate of 71.82% for standard fonts, 91% for handwritten training sets and 81.5% for testing sets. In same year Desai [3] presented OCR for handwritten Gujarati numeral recognition using a novel hybrid features extraction technique. This technique uses structural approach and statistical approach for feature extraction. Here, an image is divided into 16 sub images. Then the total no. of pixels in each sub image is found out. Thus, 16 features are extracted from the image. Using one more feature, aspect ratio, feature vector of length 17 is created for each numeral. This technique gives accuracy of 96.99% for training set and 92.78% for unseen data. Ayatullah Faruk et al. [7] presented a character segmentation technique for business card images. Images were captured by a cell-phone camera. First, an image is split into blocks that are classified as part of foreground or not based on intensity variance. Using various filters, non-text components are removed. Then text regions are extracted. Lines and characters segmentation is done using horizontal and vertical histogram profile. This technique achieves accuracy of 97.48% for character segmentation. This technique may not work well with italic or cursive text lines or words.

Baheti and Kale [8] presented work which recognized the offline isolated handwritten Gujarati numerals using affine invariant moments as feature extraction technique. Theextracted feature set is classified using Principal Component Analysis (PCA), Support Vector Machine (SVM), K-Nearest Neighbor (KNN) and Gaussian distribution function. Here, SVM gives better accuracy (92.28%) than PCA (84.1%), K-NN (90.04%) and Gaussian (87.2%) classifiers.

III. Template Creation

First, we have created templates for 10 numerals using five different fonts viz Gopika, Harikrishna, LangscapeShayamaNormal, Nilkanth, and Vakil01 as shown in figure 1. The font style used are normal, non-italics and non-bold. The font size used is 36point and 48point. To create template, standard numeral images with white foreground and black background are taken as shown in figure 2. The pixels values, which form numeral, are stored in the vector. Thus, vector represents template characteristic and is stored in the template database.



Font name	Gujarati numerals									
Gopika	૦	૧	૨	૩	૪	૫	૬	૭	૮	૯
Harikrishna	૦	૧	૨	૩	૪	૫	૬	૭	૮	૯
LangscapeShayamaNormal	૦	૧	૨	૩	૪	૫	૬	૭	૮	૯
Nilkanth	૦	૧	૨	૩	૪	૫	૬	૭	૮	૯
Vakil01	૦	૧	૨	૩	૪	૫	૬	૭	૮	૯

Figure 1. Gujarati numerals of fonts gopika, harikrishna, langscapeshayamanormal, nilkanth and vakil01

Figure 2: Image of some numerals used to create templates

IV. PROPOSED MODEL

Figure 3 shows block diagram of the proposed model.

A. *Pseudo-code of the proposed model*

The model performs the following steps:

1. Obtain grey-scale image of color image and convert it into binary image using global threshold.
2. Detect edges from the binary image
3. Detect gaps and holes from images obtained from previous step and fill the gaps and holes, if any.

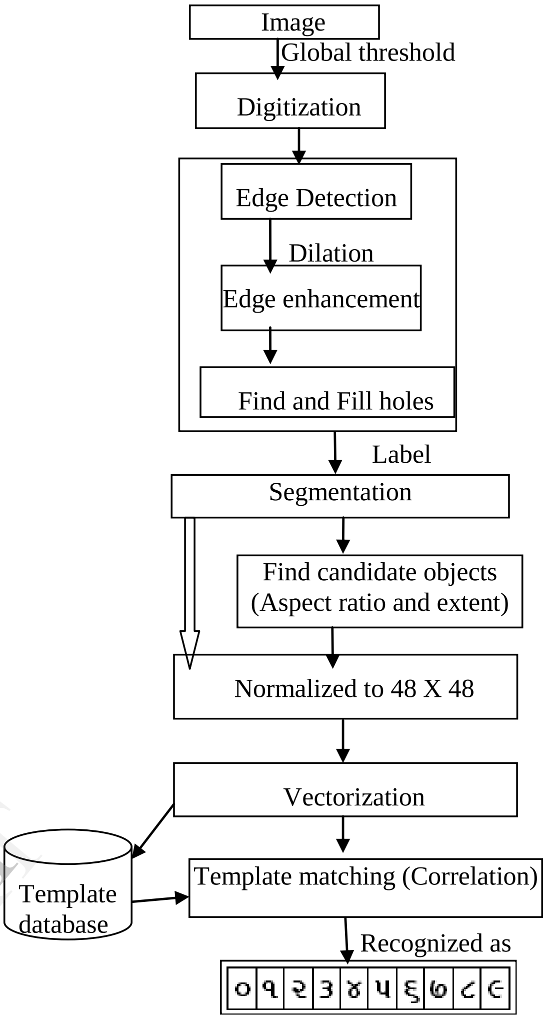


Figure 3. Block diagram of the proposed model

4. Find number of components from the image.
5. Find components having features of Gujarati numeral.
6. Extract candidate objects from grey-scale image using information of components isolated in previous step.
7. Perform normalization to put candidate objects into 48 x 48 pixels.
8. Convert pixel information of normalized image of candidate object into a vector.
9. Perform template matching for classification.

Next sections describe each step in detail.

B. *Digitization*

If image is RGB then convert it into grey-scale image. Here, grey-scale value is derived by calculating a weighted sum (W) of the R, G, and B elements using eq. 1.

$$W = wR + wG + wB \qquad \text{Eq.(1)}$$

where, $wR = 0.28989 * R$,
$wG = 0.5870 * G$,
$wB = 0.5870 * B$
W is a measure of overall luminance.

Thereafter, we have converted grey-scale image into binary image using Otsu's [9] global thresholding method.

C. *Preprocessing*

It includes edge detection, dilation and hole-filling process.

- **Edge Detection.** Edge detection process reduces data but preserves numeral as well as other content information. In this model, we have used Canny [10] method to find edges in binary image.

- **Dilation.** The edge image generated by previous step may contain gaps at edges. This means, it may contain broken edges at the boundary. This may happen because of poor binarization or noise. Therefore, it is not good to use this edge image directly for segmentation. To overcome this problem, a smoothing or gap-filling process is required.

To fill the edge gaps, we have applied morphological operation "dilation" using square structuring element with size of 2 X 2. If image of numeral is I and square structuring element S, then dilation can be defined using eq. 2.

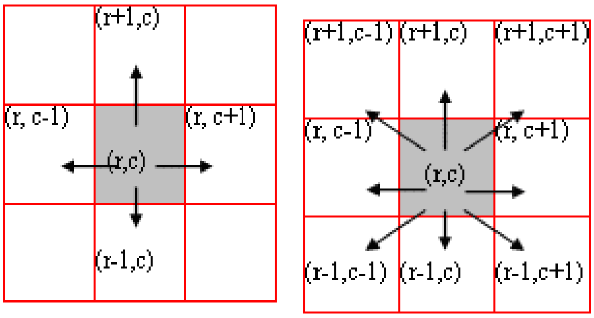
$$I \oplus S = \bigcup_{x \in S} I_x \qquad \text{Eq.(2)}$$

This process adds pixels to the edges. So, this step enhances the edges at boundary of components.

- **Hole finding and Hole-filling.** The resultant image obtained from previous step may contain extra holes, where hole is a collection of background pixels surrounded by foreground pixels. These extra holes must be filled, because filled regions are useful to locate component or object boundary. We have found these extra holes and filled them using an algorithm described in Soille [11]. This hole-filling process only fills hollow shape image components. Components having a broken edge don't change, even after hole-filling process. Thus, process preserves Gujarati numeral and non-numeral components as before.

D. *Segmentation*

It is necessary to identify each component in image. For that, we have assigned a unique label to each corresponding connected component in image using procedure described in Haralick et al. [12]. Here, pixels are labeled or grouped based on pixel neighbor or connectivity. A pixel may be connected by 4-way or 8-way as shown in figure 4.



4-way connectivity

8-way connectivity

Figure 4. Pixels connectivity

As stated earlier, all Gujarati numerals have very sharp curve, we have used 8-way connectivity.

After pixels are labeled, they are arranged into groups which make connected components. These components may be numeral(s), character(s) or any other object(s). For analysis of these components, we have found location of components and identified their bounding box (boundary). Here, bounding box is smallest rectangle that exactly fits image of segmented component. Figure 5 show segmented component with its bounding box.

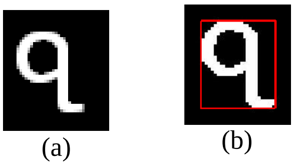


Figure 5: (a) Grey-scale image of numeral one (1)
(b) bounding box marked with rectangle

E. *Finding candidate objects for numeral*

One point is that, at this stage, Gujarati numeral components are mixed with non-numeral ones in image. There is no idea about contents of component. So, after finding components, the next step is to distinguish each component in terms of its possibility to be either Gujarati numeral or not, i.e., check whether the component is Gujarati numeral-like or not. This can be done by examining certain features of the components.

- **Noise removal.** Images may contain noise. The various reasons to introduce noise in images are quality of camera, lighting condition, quality of paper etc. There may be many small components in an image, consist of a single or few pixels. These components are too small, most likely that they are only noise and do not have any meaning at all. To remove simple noise, detail knowledge of image is not required. Here, we have filtered out components of size less than 40 pixels. The optimum value derived in a trial-and-error method is up to 40 pixels. The purpose of this process is to remove the noise as much as possible.

- **Features Extraction.** We have considered two Gujarati numeral features namely Aspect ratio and Extent.

i. Aspect ratio

The important features of Gujarati numerals can be considered are its width and height. Gujarati numerals might appear in any size in an image; therefore, it is not possible to take a decision, only on the basis of its width or height independently, whether a component is a numeral-like or not. One numeral feature, *Aspect ratio*, is obtained using both the width and the height of numeral. It is the ratio of a numeral's width to its height as given eq. 3

$$\text{Aspect ratio} = \frac{\text{Width}}{\text{Height}} \qquad \text{Eq.}(3)$$

One of characteristics of an aspect ratio is that it is independent of font size. Therefore, for any Gujarati numeral appears in any size in an image, it is expected to have constant aspect ratio. We have observed that Gujarati numerals have an aspect ratio in the range of 0.6 to 1.1. We have obtained this range by calculating aspect ratio for 10 Gujarati numeral in five fonts namely Gopika, Harikrishna, Nilkanth, LangscapeShayamaNormal, and Vakil01. The chosen font size is in the range of 22 point to 48 point. The font style used is normal, non-italics and non-bold.

ii. Extent

Another feature of Gujarati numerals is *extent* or *compactness*. An extent is the ratio of ON pixels in the region to total pixels in bounding box as given in eq. 4.

$$\text{Extent} = \frac{\text{No.of ON pixels}}{\text{Bounding box Area}} \qquad \text{Eq.}(4)$$

We have observed that Gujarati numerals have an extent value in the range of 0.25 to 0.96 (25% to 96%). We have calculated the extent for 10 Gujarati numeral in five fonts namely Gopika, Harikrishna, Nilkanth, LangscapeShayamaNormal, and Vakil01. The chosen font size is in the range of 22 point to 48 point. The font style used is normal, non-italics and non-bold. This calculation is done using output image of hole-filling process.

If aspect ratio or/and extent of components go beyond the specified range, then filtered them out. Remaining components will be Gujarati-numeral-like and considered as candidate components for numeral. Based on the knowledge of location and bounding box of the candidate components, candidate objects are extracted from grey-scale image and further processing is now performed on these extracted grey-scale candidate objects. Figure 6 shows extracted candidate object.

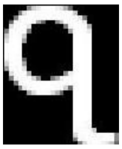


Figure 6: Extracted candidate object

The main aim of designing this step is to filter out components which do not have features of Gujarati numerals. It is useful to remove non Gujarati-numeral-like components from being consideration in recognition phase, i.e. reduces the number of objects which needs to be handled.

F. Normalization

Numerals printed on images are of different font type and size. Size of numerals also depends on how close the image has been captured by the camera. To make the model independent of font size, it is necessary to put all the candidate objects in a uniform size. So, the candidate objects extracted from the previous step are normalized to size of 48 X 48 pixels. Here, for normalization "bilinear interpolation" technique is used. Figure 7 shows normalized image of figure 6.

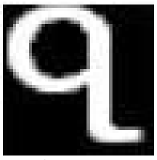


Figure 7: Image after normalization process

G. Vectorization

In order to make use of grey level intensity directly into the model, we have created vector using individual pixel information of the normalized object. The advantage of this approach is that all the relevant information is available at recognition phase.

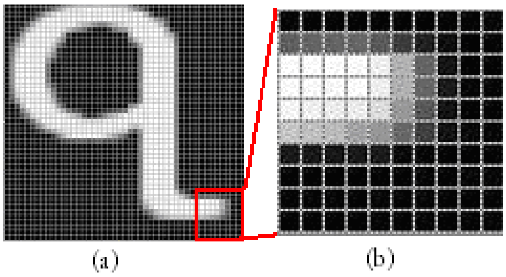
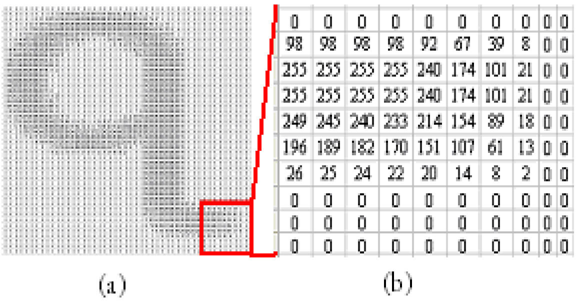


Figure 8. Grid view of normalized image shown in figure 7

Each normalized image of the candidate object is divided into 48 X 48 equal numbers of horizontal and vertical grids with 48 X 48 zones as shown in figure 8. Each zone holds one pixel with varying grey level. Now, set a value from 0 to 255 for respective zone in the grid, according to grey level. For example, set 0 for black zone and 255 for white zone as shown in figure 9. Then we have collected each column and combined together to convert the grid values into vector. This step makes a vector for normalized object of size 48X48 =

2304 as shown in figure 10.



0	0	0	0	0	0	0	0	0	0
98	98	98	98	92	67	39	8	0	0
255	255	255	255	240	174	101	21	0	0
255	255	255	255	240	174	101	21	0	0
249	245	240	233	214	154	89	18	0	0
196	189	182	170	151	107	61	13	0	0
26	25	24	22	20	14	8	2	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Figure 9. Representation of figure 8 in terms of value 0 to 255

0 98 255 255 249 196 26 0 0 0 0 98 255 255 245 189 25 0 0 0 0 98 255 255 240 182 24 0 0 0 0 98 255 255 233 170 22 0 0 0 0 92 240 240 214 151 20 0 0 0 0 67 174 174 154 107 14 0 0 0 0 39 101 101 89 61 8 0 0 0 0 8 21 21 18 13 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figure 10. Vector of numeral for image in figure 9(b)

H. *Recognition*

In this work, we have used template matching for recognition of candidate objects. Here, vector of candidate object is compared with each numeral template. A relationship is measured in terms of similarity or dissimilarity, by considering each pixel value between each numeral template Ti and the vector of candidate object. Similarity increases when a pixel in the observed candidate object is similar to the corresponding pixel in the numeral template and similarity decreases if the pixel value differs. The numeral template Ti which has the maximum similarity measure is identified and if this measure is higher than a specified threshold, then the identity of that numeral template is given to the candidate object.

We have used Correlation coefficient to measure the relationship between numeral template and vector of candidate object. The correlation coefficient (r) is calculated using eq.5.

$$r = \frac{\sum_{i=1}^{n}(Xi-\overline{X})(Yi-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(Xi-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Yi-\overline{Y})^2}} \quad \text{Eq. (5)}$$

where $-1 \leq r \leq +1$

n = No. of elements in template

Xi = ith element in template X

$\overline{X}$ = Mean of X.

Yi = ith element in vector of candidate object

$\overline{Y}$ = Mean of Y.

Different fonts represent the same numeral in different ways as shown in figure 1. Due to this, the vectors of the same numerals in different font may appear in different way. Therefore, the correlation coefficient of similar numerals varies with font name and font size. For example, table 2 gives correlation coefficient of templates of Nilkanth font with size 48 point and 36 point. Here, correlation coefficient

Table 2. Correlation coefficient of templates of nilkanth font with size 48 point and 36 point

		Size 36 point									
		૦	૧	૨	૩	૪	૫	૬	૭	૮	૯
Size 48Point	૦	**0.92**	0.08	0.23	0.37	0.17	0.17	0.16	0.53	0.25	0.52
	૧	0.22	**0.90**	0.37	0.19	0.12	0.27	0.30	0.06	- 0.07	0.19
	૨	0.30	0.24	**0.86**	0.20	0.05	0.33	0.22	0.12	0.03	0.23
	૩	0.29	0.13	0.30	**0.84**	0.20	0.19	0.08	0.09	0.10	0.03
	૪	0.16	0.08	0.01	0.39	**0.90**	0.17	0.05	0.25	0.24	0.05
	૫	0.14	0.18	0.33	0.19	0.22	**0.91**	0.36	0.06	0.00	0.03
	૬	0.12	0.30	0.23	0.13	0.01	0.28	**0.95**	0.10	0.15	0.21
	૭	0.48	0.07	0.06	0.21	0.14	0.09	0.19	**0.88**	0.20	0.25
	૮	0.10	-0.03	0.09	0.15	0.43	0.00	0.17	0.18	**0.93**	0.19
	૯	0.46	0.15	0.22	0.01	0.10	0.04	0.19	0.21	0.26	**0.95**

of same numerals having different size is in the range of 0.84 to 0.95. From table 2, we can see that relationship measure for templates of same font having different font size is not exactly 1, but closer to +1.

By experiments, we have derived that the optimum value for threshold is | 0.6 |.

V. EXPERIMENTAL RESULT

In order to evaluate and measure the accuracy of the proposed model, we have created a data set. This data set is a set of images collected from a variety of sources such as book pages, magazines, newspapers, advertisements, cover pages, etc. and were captured using digital camera. The images have characteristics like simple and complex backgrounds, with and without numerals, mixture of numerals and alphabets, with graphics and maps, textured etc. They were taken under various lighting conditions and distance. The quality of each image may differ, depending on lightning condition, the quality of the paper on which numerals are printed etc. The test set consists of 274 variable size images containing 7,079 numerals. Total numbers of numerals in the images are in the range of 0 to 290. On an average, every image has 25 (approximately) numerals.

We have implemented the proposed model using MATLAB. Figure 11 shows output of each step. The test image shown in figure 11(a) is original image and 11(b) is part of original image. It is converted into grey-scale image as shown in figure 11(c). After applying Otsu's global thresholding method to the image shown in figure 11(c), resulting binary images will look like as in figure 11(d). Figure 11(e) is an edge image. It is the result of applying the canny edge detection method to the images in figure 11(d). Figure 11(f) is the result of applying dilation process using square structuring element with size of 2 X 2. The result of applying hole-filling process to the images in figure 11(f) is

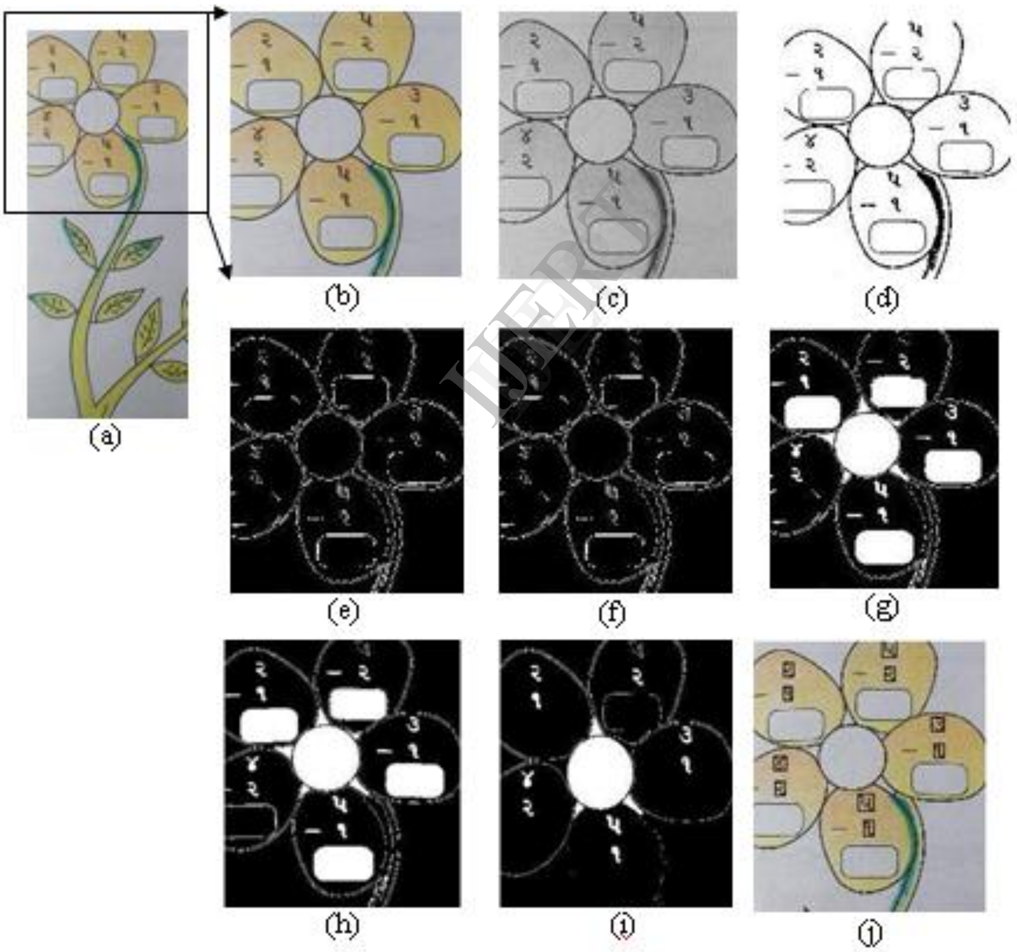


Figure 11. Output of the proposed model (a) original image (b) part of original image (c) grey-scale (d) binary image (e) edge image (d) edge image (f) after dilation (g) after hole filling process (h) after removing noise (i) candidate object (j) result of the model

displayed in figure 11(g). Figure 11(h) shows the result after removing noise. Figure 11(i) is the result obtained after applying Gujarati numeral features aspect ratio and extent. It shows the candidate objects. Finally, figure 11(j) shows the result after applying template matching process. The recognized numerals are marked with black rectangles in the original image. Figure 12 shows some experiment results on various types of images.

Let us examine the results obtained. We have applied the model on data set. Out of 7079, our model correctly recognized 6782 numerals. The model failed to recognize 297 numerals which actually exist. Thus, accuracy of 95.80% is achieved. Accuracy is calculated by dividing correctly recognized numerals by total number of numerals which are actually present as shown in eq. 6.

$$\text{Accuracy} = \frac{\text{Totalno.of numerlscorrectlyrecognized} * 100}{\text{Totalno.of numeralspresent}} \quad \text{Eq.(6)}$$

Table 3 shows the accuracy of individual numerals zero (૦) to nine (૯). Let us examine the results obtained for the different numerals. While testing, it is seen that the success rate of recognition of eight (૮) is very high, i.e. 99.21%. The success rate of recognition of numeral six (૬) is very low i.e. 77.87%. The numeral ૬ is failed to recognize 110 times out of 497, which means it is failed for 22.13%.

Although the proposed model can recognize Gujarati numerals from variety of images, it may make mistakes. In some cases, it may fail to recognize a numeral as well as misclassification also. The main reason of misclassification is the shape confusion.

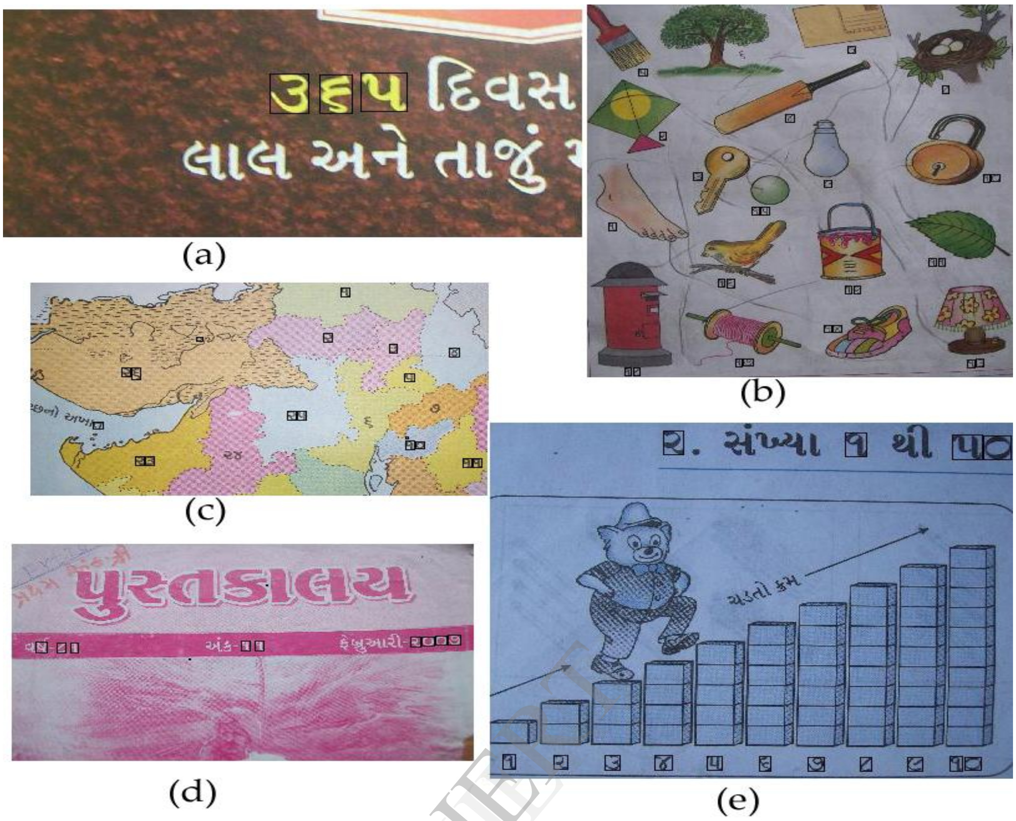


Figure 12: Result of the proposed model on various types of images (a) Texture image (b) Numerals at random location (c) Map image (d) Magazine cover page (e) Numerals are of different font size on uniform background color

Table 3. Accuracy of individual numerals zero (૦) to nine (૯)

Numeral	Total Present	Recognized Correctly	Fail to Recognize	Accuracy (In %)
૦	1007	986	21	97.91
૧	1338	1309	29	97.83
૨	915	899	16	98.25
૩	735	721	14	98.10
૪	608	602	6	99.01
૫	643	601	42	93.47
૬	497	387	110	77.87
૭	417	375	42	89.93
૮	509	505	4	99.21
૯	410	397	13	96.83
Total	**7079**	**6782**	**297**	**95.80**

To the best of our knowledge, there is no work traceable for segmentation and recognition of Gujarati printed numerals from an image. So, the results of this work could not be compared and hence we believe that this approach to segmentation and recognition of Gujarati printed numerals from image will give the best result.

VI. CONCLUSION AND PATH AHEAD

In this paper, we have discussed the significance of the work and proposed the model for segmentation and recognition of Gujarati printed numerals from an image. The proposed model is insensitive to font and style variation. Also, it doesn't depend on the size of input image. It gives accuracy of 95.80%. It works well with colored, textured, map, magazine cover pages and noisy images. This work will also be helpful for content based image retrieval.

Several modifications are possible to improve accuracy and reduce misclassification. The model is tested with off line images. It can be improved so that it works with on-line images as well. There are the shape similarities among few numerals and alphabets. Checking next character or using Gujarati language grammar may avoid misclassification. The current work can be extended in future for the recognition of Gujarati Alphabets.

REFERENCES:

[1] Karatzas, D., Antonacopoulos, A., "Two Approaches for Text Segmentation in Web Images", Proceedings of the 7th International Conference on Document Analysis and Recognition (ICDAR), Edinburgh, UK, August 2003, IEEE Computer Society Press, pp. 131-136, 2003

[2] Desai, A. A. "Gujarati handwritten numeral optical character reorganization through neural network" Pattern Recognition, vol. 43, Issue 7, pp. 2582–2589, 2010.

[3] Desai, A. A, "Handwritten Gujarati Numeral Optical Reorganization using Hybrid Feature Extraction Technique", Conf. IP, Comp. Vision and Pattern Recognition, pp. 733-739, 2010

[4] Antani, S. and Agnihotri, L. "Gujarati Character Recognition", In proc. 6th ICDAR, pp.418-421, 1999

[5] Keunhwi Koo, Jong Pil Yun, SungHoo Choi, JongHyun Choi, Doo Chul Choi, Sang Woo Kim, "Character Segmentation and Recognition Algorithm of Text Region in Steel Images", ISPRA Proceedings of the 8th WSEAS International Conference on signal processing, robotics and automation. ISPRA, pp. 293-298, 2009.

[6] Grover, S., Arora, K., Mitra, S. K., "Text Extraction from Document Images using Edge Information", IEEE India council conference, INDICON, pp 582, 2009.

[7] Ayatullah Faruk, M., Subhadip, B. and Mita, N., "Segmentation of Camera Captured Business Card Images for Mobile Devices", Journal of Computer Science and Applications, Vol. 1, Issue 1, pp. 33-37, June 2010.

[8] Baheti M. J. and Kale K. V., "Gujarati Numeral Recognition: Affine Invariant Moments Approach", 1st International Conference on Recent Trends in Engineering & Technology, Special Issue of IJECSCSE, ISSN: 2277-9477, Mar-2012.

[9] Otsu, N., "A Threshold Selection Method from GreyLevel Histogram". IEEE Trans. Systems, Man and Cybernetics, Vol. SMC-9, No. 1, pp. 62-66, 1979.

[10] Canny, J., "A computational approach to edge detection", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, pp. 679-698, Nov. 1986.

[11] Soille, P., "Morphological Image Analysis: Principles and Applications", Springer-Verlag, pp. 173-174, 1999

[12] Haralick, Robert, M., Linda, G. S., "Computer and Robot Vision", Volume I, Addison-Wesley, pp. 28-48, 1992.