ISSN: 2321-7782 (Online)
e-ISJN: A4372-3114
Impact Factor: 6.047

Volume 5, Issue 6, June 2017

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

# *Recognition of Handwritten Characters using Binarization, Image segmentation and Correlation Matching Technique*

**Monalisa Sarkar**[1]
M.Sc. Student,
Department of Computer Science,
St. Xavier's College,
Kolkata – India

**Sirsha Roy**[2]
M.Sc. Student,
Department of Computer Science,
St. Xavier's College,
Kolkata – India

**Abhirup Das**[3]
M.Sc. Student,
Department of Computer Science,
St. Xavier's College,
Kolkata – India

**Dr. Asoke Nath**[4]
Associate Professor,
Department of Computer Science,
St. Xavier's College,
Kolkata – India

***Abstract:*** ***This method presents an innovative technique to deal with the problem of offline handwritten character recognition of English characters as well as numerical and special characters. This system is very useful to the people which have a lot of paper work on a daily basis. Handwritten character recognition systems provide the solution for how to convert handwritten words into computer readable format. To attain 99.9 % accuracy in the field of HCR is very difficult. The efficiency of it depends on the classification technique used and the similarity measures taken.***

***Keywords: Character Recognition, English Alphabet Recognition, Dataset Matching, Correlation Matching.***

## I. INTRODUCTION

Handwritten character recognition is a difficult problem due to the great variations of writing styles, different size (length and height) and orientation angle of the characters. Handwritten Character recognition is an area of pattern recognition that has become the subject of research during the last some decades and has attracted many researchers across the world. Character recognition systems translate such scanned images of printed, typewritten or handwritten documents into machine encoded text. This translated machine encoded text can be easily edited, searched and can be processed in many other ways according to requirements.

Character recognition systems help humans ease and reduce their jobs of manually handling and processing of documents. Computerized processing to recognize individual character is required to convert scanned document into machine encoded form.

There are the following types of applications for handwritten recognition systems that are as follows:

1. Digital Character Conversion
2. Meaning Translation
3. Keyword Spotting
4. Signboard Translation
5. Text-to-Speech Conversion

## II. SURVEY OF TECHNOLOGIES

Handwriting recognition has been one of the most fascinating and challenging research areas in field of image processing and pattern recognition in the recent years. It contributes immensely to the advancement of an automation process and can improve the interface between man and machine in numerous applications. Several research works have been focusing on new techniques and methods that would reduce the processing time while providing higher recognition accuracy. Character recognition systems extensively use the methodologies of pattern recognition, which allots an unknown sample to a predefined class. Many techniques for character recognition are investigated by the researchers and character recognition approaches can be classified as Template matching, SVM, Neural network, Hybrid or Combination approaches and many more.

**Template matching approach**

This is the simplest way of character recognition, based on matching the stored data against the character to be recognized. The matching operation determines the degree of similarity between two vectors i.e. group of pixels, shapes curvature etc. A grey level or binary input character is compared to a standard set of stored data set. According to similarity measure a template matcher can combine multiple information sources, including match strength and k-nearest neighbour measurements from different matrices. The recognition rate of this method is very sensitive to noise and image deformation. The choice of matching depends on the nature of the image and the problem to be solved. General classifications of template or image matching approaches are: Template or Area based approaches and Feature-based approaches.

It identifies the parts on an image that match a predefined template. Advanced template matching algorithms allow to find occurrences of the template regardless of their orientation and local brightness.

Template Matching techniques are flexible and relatively straightforward to use, which makes them one of the most popular methods of object localization. Their applicability is limited mostly by the available computational power, as identification of big and complex templates can be time-consuming.

Template Matching techniques are expected to address the following need: provided a reference image of an object (the template image) and an image to be inspected (the input image) we want to identify all input image locations at which the object from the template image is present.

In *native template matching*, the search is performed in a rather straightforward way – we will position the *template* over the image at every possible location, and each time we will compute some numeric measure of similarity between the template and the image segment it currently overlaps with. Finally we will identify the positions that yield the best similarity measures as the probable template occurrences.

One of the sub-problems that occur in the specification above is calculating the similarity measure of the aligned template image and the overlapped segment of the input image, which is equivalent to calculating a similarity measure of two images of equal dimensions. This is a classical task, and a numeric measure of image similarity is usually called *image correlation*.

The fundamental method of calculating the image correlation is so called *cross-correlation*, which essentially is a simple sum of pairwise multiplications of corresponding pixel values of the images.

The correlation value seems to reflect the similarity of the images being compared, cross-correlation method is far from being robust. Its main drawback is that it is biased by changes in global brightness of the images. All that needs to be done at this point is to decide which points of the template correlation image are good enough to be considered actual matches. Usually we identify as matches the positions that (simultaneously) represent the template correlation:

- stronger that some predefined threshold value (i.e. stronger than 0.5)
- locally maximal (i.e. stronger than the template correlation in the neighboring pixels)

There are some advanced methods in this category. These are GrayScale-based Matching, Edge-based Matching.

**Neural Networks**

Various types of neural networks are used for character recognition classification. A neural network is a computing architecture that consists of massively parallel interconnection of adaptive neural processors. Because of its parallel nature, it can perform the compared to classical techniques. Because of its adaptive nature, it can adapt to changes in the data and learn the characteristics of input signal. Output from one node is fed to another one in the network and final decision depends on the complex interaction of all nodes. Several approaches exist for training of neural networks like error correction, Boltzman, Hebbian and competitive learning. Neural network architectures can be classified as, feed-forward, feed-back and recurrent networks.

The neural network is presented with a target vector and also a vector which contains the pattern information. Then it attempts to determine if the input data matches a pattern that the neural network has memorized.

A feed forward back propagation neural network having two hidden layers is used to perform the classification. The hidden layers use log sigmoid activation function, and the output layer is a competitive layer, as one of the characters is to be identified. The feature vector is denoted as X where X = (f1, f2,…,fd) where f denotes features and d is the number of zones into which each character is divided. The number of input neurons is determined by length of the feature vector d. The total numbers of characters n determines the number of neurons in the output layer. The number of neurons in the hidden layers is obtained by trial and error.

**Support Vector Machine**

Support vector machine was proposed by Vapnik as binary classifier. It represents one of the latest supervised learning classifiers and it was used in numerous applications. SVM discovers a hyper-plane that separates data from different classes. Each instance is labelled with one of existing classes and they are represented as points in space. SVM builds a model based on instances from training set and further classification of unknown instances is done by that model. Hyper-plane separates the labelled instances from the training set. This hyper-plane is determined by the nearest instances that are called support vectors. Hyper-plane should be as far as possible from instances of both classes.That means the hyper-plane provides the highest margin distance between the nearest points of the two classes (called functional margin). This approach, in general, guarantees that the larger the margin is the lower is the generalization error of the classifier.

By nature SVMs are essentially binary classifiers, however, based on several researchers' contributions they were adapted to handle multiple class cases. The two most common approaches used are the One-Against-All and One-Against-One techniques, but this scenario is still an ongoing research topic.

## III. METHODS USED IN RECOGNIZING HANDWRITING CHARACTERS

The approach used in this paper is summarized as below:

Firstly handwritten texts are to be scanned using digital camera or a scanner and create a digital jpeg image. The digital image is read by an inbuilt function and then takes the whole image in a 2d array form where each element is the value of each pixel. Then it performs rgb2gray and then gray2binary operation to convert all the pixel value to 1 or 0. That means it takes the drawn image pixels as value 0 and other white pixels as 1. After the binarization operation it selects only the drawn image from the original image by eliminating white part in 4 sides using edge detection and segmentation operation. Then the first character is used for match operation. And the result of it is stored in a separate file. After matching the first character image, again it takes one character from the rest of the image and these operations are performed until no character remains in the same row of that image.

Before the matching it first resizes both the character image and dataset image in same resolution otherwise the match operation can give erroneous result. It is done by using a MATLAB inbuilt function imresize.

IMRESIZE (A, [NUMROWS NUMCOLS],'nearest') resizes the image so that it has the specified number of rows and columns. Either NUMROWS or NUMCOLS may be NaN, in which case IMRESIZE computes the number of rows or columns automatically in order to preserve the image aspect ratio. 'nearest' is used for nearest-neighbour interpolation.

Following are the operations performed on the input image:


Fig 1: Step-wise procedure

***Image Acquisition:*** First a scanned image of handwritten text is acquired using a digital camera or a scanner and a digital jpeg image is created on which the following procedures are applied.

***Image Pre-Processing:*** This consists of two processes: (i) conversion of RGB to Grayscale image and then (ii) conversion of Grayscale to Binary i.e. Binarization.

In *RGBtoGray* conversion, it takes the original scanned image matrix as input and processes each pixel in such a way that it results in a gray scale image, where the pixel values lie in between 0 to 255.

In *Binarization*, it takes the gray scale image matrix as input and processes each pixel of it in such a way that, the drawn image pixels are represented by 0 and other white pixels are represented by 1.

***Character Edge detection*:** In this phase, the boundary of each character of the binary image is detected. It detects the top row, bottom row, left column and right column boundaries of each character.

The edge detection process starts with left and right boundary detection of the first character. The system start scanning the segmented image column wise, as soon as it gets a black pixel in a column, it returns the column index as the left boundary edge of the character. Then for right edge detection, the scan starts from the left boundary and continues in the same columnwise manner. As soon as the system finds a column with all white pixels, it consider it as a space between two consecutive character and returns the pervious column index as the right boundary edge of the character. The resultant image is stored in a separate file by eliminating the portion on the left side of the left boundary edge and the portion on the right side of the right boundary edge.

Next, for top and bottom edge detection the system starts scanning the resultant image row by row. Starting from the top left corner of the resultant image, as soon as it gets a black pixel in any row, it designate it as the top edge of the character. Then for bottom edge detection, the scan starts from the last row of the image and as soon as it gets a black pixel in a row, it designated that row as the bottom boundary edge.

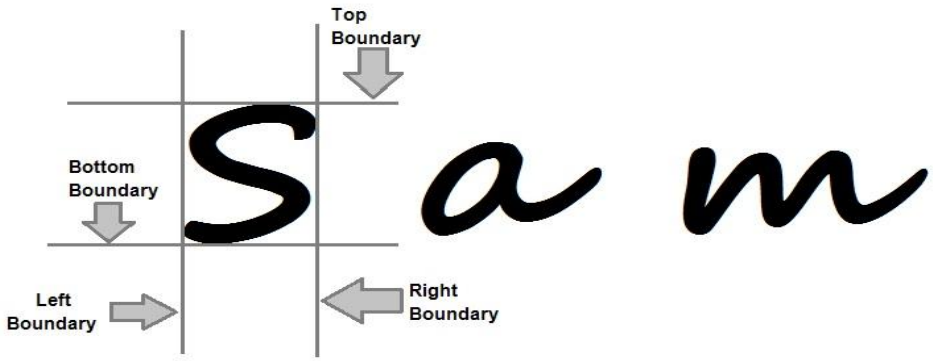



Fig 2: Character boundary detection

***Character Segmentation:*** This process mainly consists of extracting each character as per the detected edges by scanning each row and column from the top left corner of the image and storing them into another 2D matrix of pixels. This 2D array is then checked against the database images for matching.

***Character Matching:*** In this phase, the matching operation between the segmented character image and each of the images in the dataset created is performed.

In this matching algorithm, the correlation co-efficient (which typically ranges between -1 and 1 in its normalized form) between the acquired character image and every image from the dataset is calculated. The correlation co-efficient is calculated based on Pearson's method of correlation.

Correlation is used for finding matches of a sub-image w(x,y) of size JxK within an image f(x,y) of size MxN where we assume J<=M and K<=N. Correlation coefficient is calculated by the following formula :

$$Rho=cov(A,B)/(Std(A)*Std(B))$$

Where cov(A,B) is the covariance of A and B which is calculated as : average(A1*B1) , where A1=A-A' [A' is the mean value of the pixels in A]. B1=B-B' [B' is the mean value of the pixels in B].

And std(A), std(B) is the standard deviations of A and B.

***Character printing:*** Along with the dataset of images there is also a data array of characters that is maintained in the exact order as of the dataset images. The character with the maximum value of the correlation coefficient is recognized as the identified character and its respective index from the data array is taken which is then printed on the screen.

The dataset has been created by some pre-handwritten samples that have been obtained from various sources. This dataset is the entity with which the input image will be matched and the result will be produced accordingly. Some of the samples of the dataset are illustrated below:

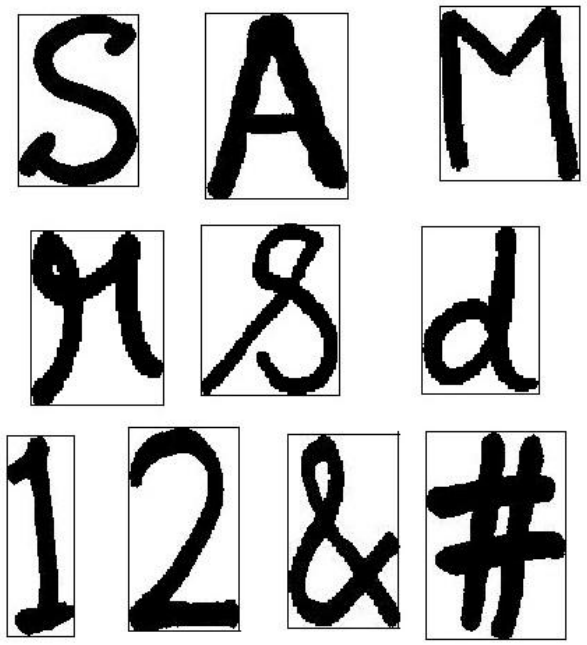


Fig 3: A few samples from the dataset

## IV. ALGORITHMS

(i) Binarization: Conversion from grayscale to binary image. Black pixels are denoted by the value 0, and white pixels are denoted by 1.

Step 1: Accept the grayscale image matrix as input.

Step 2: Calculate the height and width.

Step 3: i ← 1.

Step 4: j ← 1.

Step 5: if $pixelvalue_{i,j}$ >= 127 then goto step 6 else goto step 8

Step 6: pixelvalue ← 0.

Step 7: goto step 9.

Step 8: pixelvalue ← 1.

Step 9: j ← j + 1.

Step 9: If j < width of matrix , go to step 5

Step 10: i ← i + 1.

Step 11: If i < height of matrix, go to step 4

Step 12: End.

(ii) Edge detection: This algorithm contains two parts which are to be performed one after the other to get the desired result.

*Left and right edge detection.*

Step 1: Start.

Step 2: Accept the image A.

Step 3: Calculate height and width of A.

Step 4: j ← 1.

Step 5: i ← 1.

Step 6: if $A_{ij} = 0$, the left most point of the character has been identified.

left_edge ← i and goto step 11.

Step 7: i ← i + 1

Step 8: if i < height goto step 6

Step 9: j ← j + 1

Step 10: if j < width goto step 5

Step 11: Now we begin scanning from the left_edge.

Set i ← left_edge.

Step 12: Set j ← 1

Step 13: If for one column, all $A_{i,j} = 255$, right_edge ← i – 1 and goto step 18.

This all white column indicates the white space between characters.

Step 14: i ← i + 1

Step 15: if i < height goto step 13

Step 16: j ← j + 1

Step 17: if j < width goto step 13

Step 18: Crop out the character with respect to its left_edge and right_edge.

Step 19: Stop.

*Top and bottom edge detection.*

Step 1: Start.

Step 2: Accept cropped out image A from the previous algorithm.

Step 3: Calculate height and width of A.

Step 4: i ← 1

Step 5: j ← 1

Step 6: If $A_{i,j} = 0$, the top most point of the character has been identified.

top_edge ← i and goto step 11

Step 7: j ← j + 1

Step 8: if j < width, goto step 6

Step 9: i ← i + 1

Step 10: if i < height, goto step 5

Step 11: i ← height.

Step 12: j ← 1.

Step 13: if $A_{i,j} = 0$, bottom_edge ← i and goto step 18.

Step 14: j ← j + 1

Step 15: if j < c goto step 13

Step 16: i ← i - 1

Step 17: if i > 1 goto step 12

Step 18: Stop

(iii) Matching

Step 1: Start.

Step 2: Set B to be the image to be matched and the data set considered is D. D contains the images of the characters with which the image B will be matched.

Step 3: B is matched with each element of D; $D_i$ denotes each element of D, where i := 1,2,3…(length of D).

amt ← match(B, $D_i$)

amt is a value returned by the match function. The value of amt is high if B and $D_i$ are very similar.

Step 4: if amt is maximum for a specific (B, $D_i$) pair, $D_i$ is considered the best match for B.

Step 5: Stop.

match(A,B) – this function accepts two image and matches them according to correlation coefficient matching technique. match(A,B) will return the highest value if A is the best match of B within the whole data set.

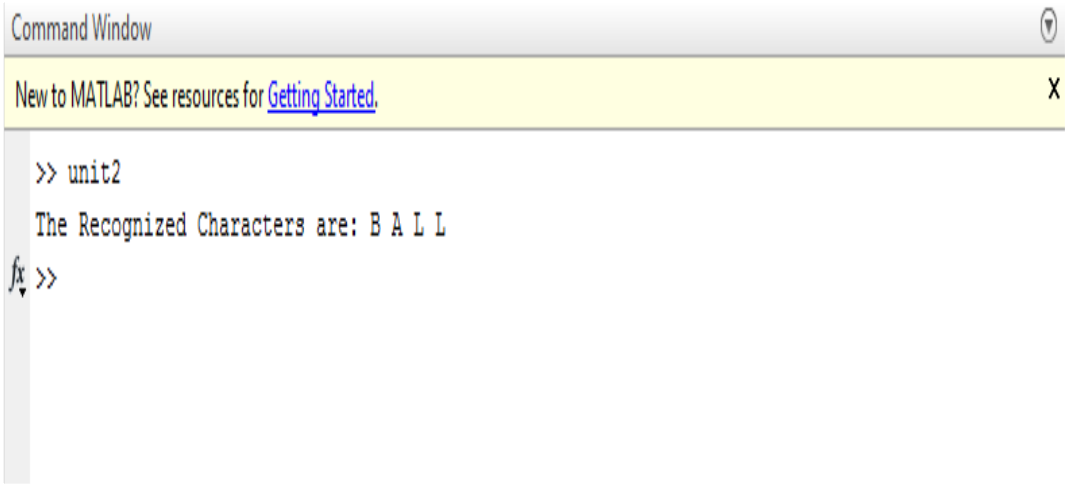
## V. RESULTS AND DISCUSSIONS

A few cases of the inputs provided and the corresponding outputs obtained are illustrated below:

Set1:

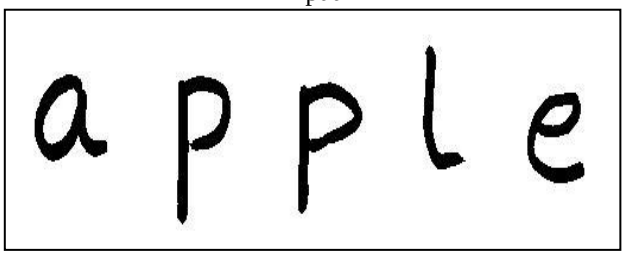
Input

Output

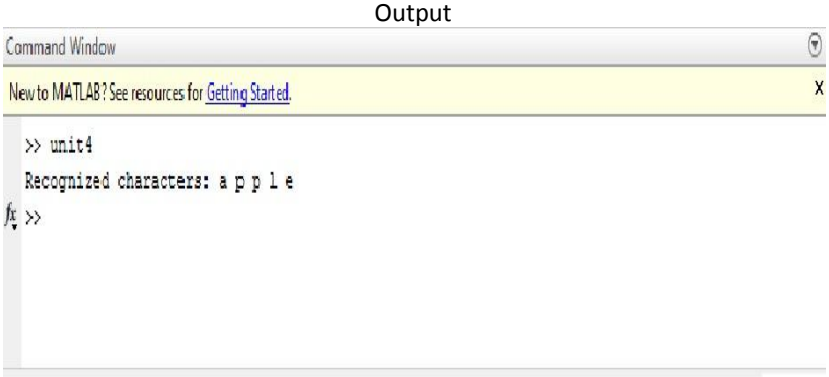



Set 2:

Input

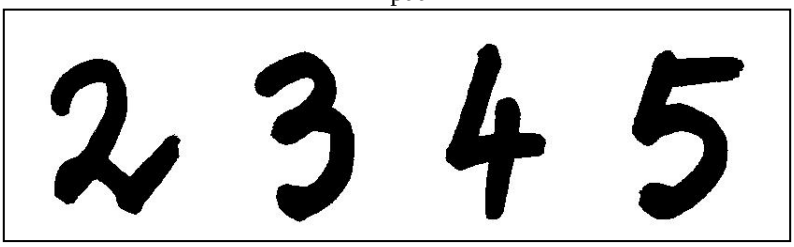


Output

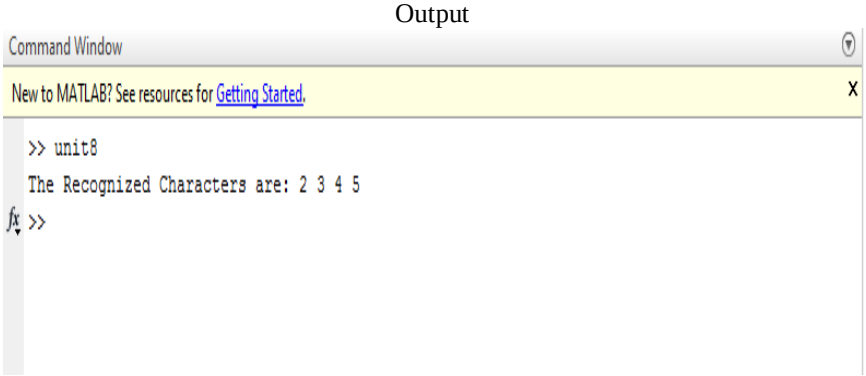



Set 3:

Input



Output



## VI. Limitations of the Presented Methods

(i) The present work is comparatively slow.

(ii) The present work cannot be applied for cursive writing because between each consecutive character a space (a column with all white pixels) is considered for segmentation.

## VII. Conclusion and Future Scope

A simple and effective template matching method for identification of handwritten characters was introduced. Template matching is the common method of character recognition techniques. The characters were extracted from input image and normalized. For recognition process, the extracted character was compared to each template in the dataset to find the closest representation of the input character. The matching metric was computed using 2-D correlation coefficients approach to identify similar patterns between the test image and the dataset images. Experimental results show that the proposed method is efficient for identification handwritten characters. Initially, the work progressed with a native template matching approach i.e. pixel-by-pixel method. Due to unsatisfactory result in terms of accuracy and time consumption, finally an improved template matching approach i.e. matching by image correlation has been followed. The proposed method has been applied on different unknown characters. It gives the 80% accuracy.

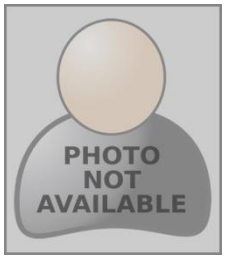
The future scope includes recognition of cursive handwriting and online character recognition.

## References

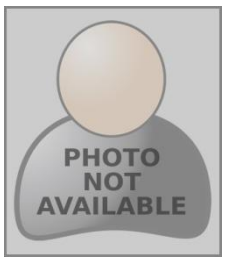
1. Monish Dutta, Pritha Roy, Sunanda Datta, Asoke Nath -"A new method for Recognition of Handwritten characters using edge-detection, segmentation, pattern matching" ,International Journal of Innovative Research in Computer and Communication Engineering (IJICCE) , Vol-3, Issue-6,June 2015.
2. Anshulgupta and Manisha srivastava , "offline handwritten character recognition" , Department of electronics and communication engineering, Indian Institute of Technology ,Guwahati, Assam, India – 781039, April, 2011.
3. Neeta Nain Subhash Panwar ,"Handwritten Text Recognition System Based on Neural Network", Malaviya National Institute of Technology Jaipur, Neeta Nain Subhash Panwar.
4. Hassoun, M.H., "Fundamentals of Artificial Neural Networks", MIT Press, Cambridge, London, England,1995.
5. AviDrissman, Dr. Sethi, CSC 496, February 26, 1997, "Handwriting Recognition Systems: An Overview".
6. MANSI SHAH AND GORDHAN B JETHAVA, Department of Computer Science & Engineering Parul Institute of Technology, Gujarat, India.Information Technology Department Parul Institute of Engg. & Technology, Gujarat, India, "A LITERATURE REVIEW ON HAND WRITTEN CHARACTER RECOGNITION".

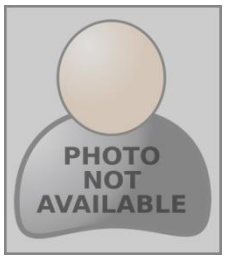
## AUTHOR(S) PROFILE




**Monalisa Sarkar,** is a student of M.Sc. (2015-2017), Department of Computer Science, St. Xavier's College (Autonomous), Kolkata.




**Sirsha Roy,** is a student of M.Sc. (2015-2017), Department of Computer Science, St. Xavier's College (Autonomous), Kolkata.




**Abhirup Das,** is a student of M.Sc. (2015-2017), Department of Computer Science, St. Xavier's College (Autonomous), Kolkata.

**Dr. Asoke Nath**, is Associate Professor in Department of Computer Science, St. Xavier's College (Autonomous), Kolkata. He is involved in research areas like Cryptography and Network security, Visual Cryptography, Steganography, Green Computing, MOOCs and e-learning, Social Networks, Big data etc. He has published more than 216 research papers in International and National Journals and International Conference Proceedings.