Grenze International Journal of Engineering and Technology, January Issue

Enhancing Security: Devanagari Handwritten CAPTCHA Generation using Digital Handwritten Devanagari Character Dataset

Sanjay E. Pate[1] and Prof. Dr. Rakesh J. Ramteke[2]
Asst. Prof., Nanasaheb Y. N. Chavan Arts, Science & Commerce College, Chalisgaon
E-mail: sanjayepate@gmail.com
Senior Professor, School of Computer Sciences, Kavayitri Bahinabai Chaudhari North Maharashtra University, Jalgaon, Maharashtra
E-mail: rakeshramteke@yahoo.co.in

***Abstract*—CAPTCHA, Automated Public Turing Tests, serves as a critical mechanism for human and machine identification online. However, the prevalence of primarily English text-based CAPTCHA poses a significant barrier for individuals in rural India. To address this challenge, we introduce a new Devanagari script text-based CAPTCHA designed to increase accessibility for users in diverse linguistic contexts. Our approach involves the creation of a Devanagari handwritten dataset, consisting of 44,000 images featuring 34 alphabets and 10 numerals. Each grayscale character has a resolution of 65 by 65 pixels. The dataset is publicly available on IEEE and Mendeley data ports. The algorithm for Devanagari Handwritten CAPTCHA generation is implemented on Python's Anaconda Jupiter platform, adhering to CAPTCHA standard guidelines. This includes image resizing, enhancement, wrapping, dilation, and the introduction of background noise such as arcs, lines, dots, and clutter with Gaussian, Salt and Pepper, Poisson, and Speckle noise to bolster resistance against image-breaking techniques. Each CAPTCHA image is standardized at 250 X 90 pixels, resulting in the generation of nine distinct CAPTCHA types, with 10,000 images of each type, contributing to a dataset of 90,000 images. This dataset, now available on Mendeley Data, serves as a valuable resource for researchers to test and enhance CAPTCHA recognition systems in a diverse linguistic landscape.**

***Index Terms*— CAPTCHA, Devanagari, HCI, Bot.**

I. INTRODUCTION

The Devanagari script, a phonetic system with origins in the ancient Brahmi alphabet, forms the foundation for numerous Indian languages. According to data from 2023, approximately 342 million individuals speak Hindi written in Devanagari script, making it the fourth most widely spoken language among the top 45 globally. The Devanagari script comprises roughly 34 consonants and 14 vowels, with no distinction between uppercase and lowercase letters, written left to right.[1] Internet bots are software programs that execute automatic operations over the Internet using scripts. CAPTCHA is a mechanism designed to differentiate between human users and computer bots. It achieves this by employing automated public Turing tests to identify machines and humans to prevent abuses. Research has been conducted on non-English language CAPTCHAs, and a methodology for

Grenze ID: 01.GIJET.11.1.532

designing handwritten CAPTCHAs has been proposed. The Handwritten Devanagari CAPTCHA is an example of this, and its algorithm and experimental analysis are detailed in Section 5. The article concludes with the key findings and inferences derived from this research.

II. Review of Literature

A. Non-English Language CAPTCHA Research Papers Review

1. Researchers Asadullah Kehar and Rafaqat Hussain Arain, along with their colleagues, designed Sindhi text-based CAPTCHAs for regional websites, which are comparable to the Arabic language CAPTCHAs. These CAPTCHAs are composed of a sequence of 62 characters, incorporating colored ellipses, clutter, and dots. The CAPTCHA was developed using C#, and the characters were overlapped to prevent CAPTCHA OCR programs from segmenting the string. The web page was created using ASP and JSP, and it was evaluated by a group of users.[19]
2. In the year 2007,M. Hassan Shirali-Shahreza and Mohammad Shirali-Shahreza unveiled the concept of a "Multilingual CAPTCHA," which is characterized by presenting users with a grid of images and requesting them to select the appropriate one. This method employs native language messages, thus preventing computerized attacks. The creators implemented this technique utilizing PHP, making it compatible with a range of devices, such as mobile phones, PDAs, and touch screens. This approach is comparable to an implicit CAPTCHA and has demonstrated effectiveness in combating automated attacks. [2]
3. In 2008, M. Hussen Shirali-Shahreza introduced the "Advanced Nastaliq CAPTCHA," a method that incorporated Arabic and Persian characters, utilizing the Nastaliq font and PNG format. This technique did not involve adding noise or distortion to the image, as the letters were connected, and over half of them contained dots. Additionally, the text was written in a right-to-left format, thereby making it difficult for optical character recognition (OCR) programs to identify words without the inclusion of noise or distortion. The implementation of the Advanced Nastaliq CAPTCHA was achieved using Java.[3]
4. Sushma Yalamanchili and Kameswara Rao created a text-based CAPTCHA system called DevaCAPTCHA for Indian languages in the year 2011. This framework consists of a database of Devanagari script text, a query generator, an obfuscator to distort and add noise to text, an interface, and a match response. The obfuscator employs patterns such as mosaics, arcs/jaws, and vertical overlapping to confuse the system. Future research will focus on the implementation of DevaCAPTCHA and optical character recognition (OCR) testing for Indian language scripts, as well as Devanagari handwriting recognition.[4]
5 In 2012, S. Ravi Kiran and Y. Rama Krishna put forth a fresh perspective on shielding users' passwords from spyware attacks by integrating CAPTCHA and graphical passwords. The goal of this study is to recognize registered CAPTCHA images and fortify the security of graphical passwords, all while enhancing login time and promoting memorability.[6]
6. Bilal Khan et al. investigated a novel approach to internet security in their 2013 study, "Cyber Security Using Arabic CAPTCHA Scheme." This approach incorporates an Arabic-based scheme and involves a database of 2,71,000 Arabic words and techniques such as random character selection, pre-processing, segmentation, and character recognition to achieve high readability rates. The authors' proposed scheme is a promising solution for application in non-Arabic-speaking countries. [7].
7. Baljit Singh Saini and Anju Bala introduced a captcha system utilizing the Gurmukhi script in 2013, which employed an attractive image created from a sequence of Punjabi characters. This system aimed to protect bots by employing a technique that involved adding noise dots to the image through the manipulation of both foreground and background colors. The implementation of this scheme was accomplished using C# and ASP.NET, which garnered an average success rate of 75%. Moving forward, the potential future scope of this research may involve exploring handwritten Punjabi captchas.[8]
8. M. Tariq Bandy and Shafila Afzal Sheikh formulated a model aimed at tackling Indian regional multi-lingual CAPTCHA challenges, which encompass 22 official languages in the country. The model involves obtaining the language, fetching Unicode characters, generating a string, adding noise, storing the string, and comparing test results.
Further research may potentially explore the use of handwritten strings or words.[9]

B. Generation of Captcha Research Papers Review

The paper by Yadav, Agarwal, Uprety, and Batham introduces a novel approach for generating random CAPTCHA text that incorporates the current date and time. This system retrieves the present date and time, extracts the digits, performs arithmetic operations, generates code using 62 ASCII English characters while

eliminating repetition, and produces random text that is difficult for unskilled users and standard code to decipher.[10]

S. A. Alsuhibany and M. T. Parvez have developed a technique for generating Arabic Handwritten CAPTCHAs using existing Arabic images from the KHATT database. This approach involves segmentation, transforming Persian Arabic Wei (PAW) images into binary format, and determining the baseline. This method addresses challenges such as distorted characters, random colors, noise, and broken characters. The performance of the CAPTCHAs is assessed using a comprehensive word recognition system, resulting in a usability accuracy of over 88%. [12]

The research conducted by Tang, Gao, Zhang, Liu, Zhang, and Wang employs deep learning techniques to examine text-based CAPTCHA images from the top 50 websites in English, Chinese, Arabic, and Roman languages. The study introduces an innovative image-based CAPTCHA, SACaptcha, which focuses on semantic information interpretation and pixel-level segmentation instead of image classification, as opposed to earlier image-based CAPTCHAs. The primary goal of this investigation is to improve the security measures in CAPTCHAs. [14]

A recent study by P. Panwar, Monika, P. Kumar, and A. Sharma has unveiled a novel approach for generating CAPTCHAs through the application of hand gesture recognition techniques. This method involves presenting a form with a CAPTCHA image and a message instructing the user to mimic the displayed gesture. The model then verifies whether the user's gesture corresponds to the one shown, thereby confirming user identification. This approach employs image conversion, morphological filtering, and edge detection, with an accuracy rate of 80%. [16]

Mohinder Kumar and Manish Kumar Jindal's work delves into the development of a secure Devanagari CAPTCHA for the (http://www.captcha.com) website. The researchers selected 20 unique designs from a total of 60, employing 39 Devanagari characters. The designs were successfully separated and denoised, resulting in high success rates ranging from 88.14% to 98.06%. However, the current design is susceptible to bots that employ unsecured methods. To tackle this issue, the authors recommend incorporating a background, a more extensive character set, handwritten characters, color, character thickness, overlapping, scaling, font, and noise patch.[18]

III. Problem Statement

Why DEVANAGARI CAPTCHA?

In the year 2023, Devanagari script spoken by more than 342 million individuals worldwide, held the fourth position among the top 45 languages.[1].

The English-based CAPTCHA script is challenging for rural people speaking local languages, as most government websites in India provide information in Devanagari, but Devanagari CAPTCHAs are not used on these websites.

Around **1,36,360** websites in India utilize CAPTCHA in English, with **99%** using Google's reCAPTCHA and **1%** using another service. (hCaptcha, Sweet Captcha, Math Captcha, Snap host, WP Captcha, SI Captcha, Captcha, Human Presence, Friendly Captcha) [5].

Here our objective is: To design and implement an algorithm for Handwritten Devanagari CAPTCHA generation, using our own designed Devanagari Handwritten character and digit dataset of 44,000 images which follows the Standard guidelines of CAPTCHA design.

IV. Proposed Methodology

A. Dataset

Used our own designed “Handwritten Devanagari Character Data Set of 44,000 Images for Devanagari CAPTCHA generation”.[26][27]

B. Image processing techniques used

This technique involves modifying images, obtaining enhanced images, or extracting valuable information for future use.

i.Resizing image: Image interpolation involves resizing or distorting images from one-pixel grids, while picture scaling is needed to increase or decrease the total number of pixels. Matlab and Python programs can be used to reduce printed and handwritten alphabet images to 65 x 65 pixels.

ii. Types of Noise in CAPTCHA Images:

CAPTCHA images often feature added noise to improve the performance of the Optical Character Recognition (OCR) system. This added noise, referred to as CAPTCHA noise, comes in various intensities and forms, and can be represented by the equation: A(x,y) = H(x,y) + B(x,y).

In this equation, A(x,y) represents the noisy image, H(x,y) represents the function of image noise, and B(x,y) represents the function of the original image. CAPTCHA noise is randomly added to the image.

Different types of Noise in the image:1.Gaussian Noise refers to a statistical noise with a probability density function akin to a normal distribution, which is generated by adding a random Gaussian function to the Image function. This type of noise is commonly used in image processing and analysis.[15]

2Salt and Pepper Noise, also known as data drop noise, is a technique used to modify images by adding random bright and dark values, statistically dropping the original data values. This type of noise is often used to simulate real-world conditions.[15]

3. Poisson Noise, also known as quantum or shot noise, is a type of noise that is produced by electromagnetic waves, such as x-rays, visible light, and gamma rays. This noise is caused by the random fluctuation of photons and is a significant issue in medical imaging systems.[15]

4. Speckle Noise is a granular noise that is commonly found in optical and digital holography. It is a significant issue in image reconstruction due to its natural occurrence and its reduction in quality due to the multiplication of random pixel values with distinct picture pixels.[15]

In summary, noise is a common problem in image processing and analysis. There are several types of noise, including Gaussian Noise, Salt and Pepper Noise, Poisson Noise, and Speckle Noise. Denoising methods are often used to restore the original image and improve the quality of the image.

C. Guidelines for Generating CAPTCHAs: [17]

1. The Character Set: The Devanagari CAPTCHA makes use of a set of 34 characters, inclusive of 10 digits, resulting in a total of 44 characters. The dataset under consideration consists of 44,000 images of alphabets and digits.

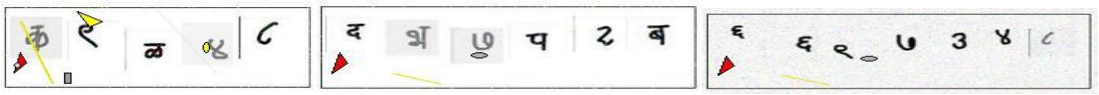
Numerals (10)	Vowels (04)	Consonants (30)
० १ २ ३ ४ ५ ६ ७ ८ ९	अ इ उ ए	क ख ग घ च छ ज झ ट ड ढ ण त थ द ध न प ब भ म य र ल व श ष स ह ळ

Figure- I - Selected Devanagari Character Set

2. Character set: One of the distinctive aspects of this technique is the utilization of a digital handwritten character set.

Figure II : Devanagari Characters from Dataset

3. Length of CAPTCHA String: The CAPTCHA string has a variable length, usually ranging from 5 to 7 characters. At runtime, a random pick is made.

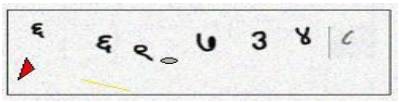


4. Revised Text: Adjusting Character Dimensions:

TABLE I: ALTERING SYMBOL IMAGE SIZE WITH CAPTCHA STRING LENGTH

CAPTCHA length = 5	CAPTCHA length = 6	CAPTCHA length = 7
54 X 54 Pixel	45 X 45 Pixel	39 X 39 Pixel

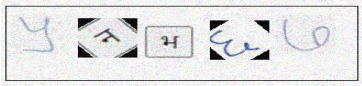
5. Font Selection: The dataset is handwritten, resulting in each image having a unique shape and size.
6. Scaling: The act of adjusting the dimensions of a digital image through the application of bicubic and bilinear interpolation techniques entails the creation of both smaller and larger images, with the scale range encompassing values between +5 and +10.



7. Utilizing Bicubic 300 and Bilinear -400 for rotation.

The scale and rotation of Devanagari character images in CAPTCHA systems can have an adverse effect on the usability of optical character recognition (OCR) technology.
8. Add geometric symbols such as lines, arcs, polygons, and rectangles to CAPTCHA images.:

9. Noisy Patch: Noise refers to an unpredictable change in the intensity or hue of an image.
i. Adjust image contrast by either enhancing or diminishing it.
ii. The CAPTCHA image was randomly activated with one of the following different types of noise: Gaussian, localvar, Salt and Pepper, Poisson, or Speckle.



V. Experimental Analysis and the Outcomes

A. Algorithm for CAPTCHA generation

1. Choose one set at random from the following three combinations: Handwritten Character (HC), Handwritten Digit
(HD), or a combination of both (HC-HD).
2. Configure the directories for the CAPTCHA images and character set.
3. Adjust the character set's length to match the above option (34 characters for HC, 10 characters for HD, and 44
characters for HC-HD).
4. Enter the number of CAPTCHA pictures needed (L).
5. For each CAPTCHA picture (CI = 1 to L), repeat steps 6 through 19.
6. A read file containing an image of a square-shaped box that generates CAPTCHA text.
7. Choose the CAPTCHA image's length at random (5, 6, or 7 characters) and assign it the variable CL.
8. Select a directory or folder to hold the CAPTCHA image.
9. For each character in the CAPTCHA image (I = 1 to CL), repeat steps 10 through 14.
10. Select a Devanagari alphabet or digit at random from a total of 44 options.
11. Choose one image from a set of 1000 images labeled "K" from a specific folder.
12. Adjust the size of the selected image by the value of CL:
a. Resize to 54 x 54 if CL is equal to 5.
b. Resize to 45 x 45 if CL is equal to 6.
c. Resize to 39 x 39 if CL is equal to 7.
13. Undertake any of the five available operations at random on the selected image to introduce imperceptible noise to
OCR. The following options are available:
a. Rotate the image by 90 degrees in a bicubic manner.
b. Rotate the image by -40 degrees in a bilinear direction.

c. Enlarge the image by +5 points.
d. Reduce the image's contrast.
e. Strengthen the contrast of the image.
14. Place the selected image in a square-shaped box at a predetermined location.
15. Adjust the size of the image to fit within the square-shaped box.
16. Save the newly constructed CAPTCHA image in a BMP, JPG, or JPEG file format.
17. Introduce noise into the image using the following parameters: "Gaussian", "localvar", "Poisson", "salt", "pepper",
"s&p", or "speckle".
18. Insert a line, circle, polygon, or ellipse within the image as a noise.
19. Store the CAPTCHA image in the appropriate folder.

B. Algorithm to test the Devanagari CAPTCHA image at the user interface

1. During the CAPTCHA image-generating process, a set of Unicode values for each symbol is acquired and saved in an array.

2. The Unicode values of each symbol supplied by the user via a virtual keyboard are saved in an array as enter the string.

3. Next, the function assesses whether every element in the two arrays is the same by comparing them. The function will return "string match." if every element matches.

C. Implementation

Experimental Environment
The present undertaking employs Python on the Jupyter platform within a Windows environment, featuring a 64-bit Windows operating system. The necessary hardware specifications comprise an Intel Core™ i5 processor, 8GB RAM, and a 4GB NVIDIA GEFORCE GTX GPU. Python has been selected owing to its more rapid execution time in comparison to Matlab. Python libraries incorporated in this project encompass NumPy, Pandas, Scikit-learn, TensorFlow, Keras, OpenCV, PyGame, PyTorch, and Tesseract OCR. OpenCV functions are utilized for image manipulation.

D. Outcome

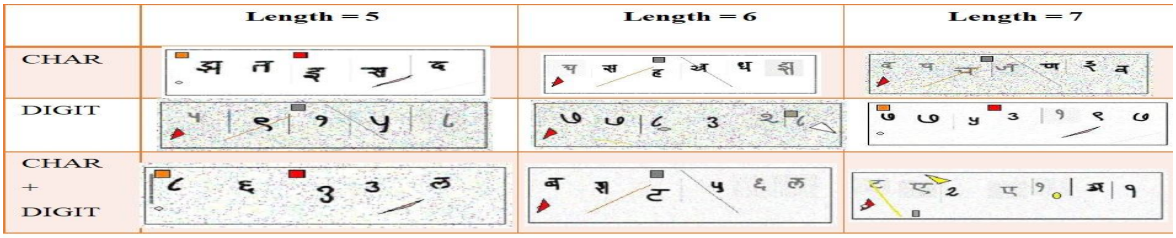


Figure-1. Devanagari Captcha Images

The dimensions of the CAPTCHA images are 250 x 90 pixels, by standard generating rules. There are two different character sets used: Digit and Handwritten Alphabet. These sets yield three classes: digit, char, and a mixture of digit and char. Three subclasses of length FIVE, SIX, and SEVEN (5, 6, 7) exist for each class. Nine subclasses are produced as a result (3 classes x 3 subclasses).
To put it simply, images are produced for every class in the chosen character set. Ten thousand CAPTCHA images are made for each class. Thus, using Python, a dataset of 90,000 (ninety thousand) images is created for the nine classes. The Python programming language's Jupyter platform and Anaconda are used to implement Devanagari CAPTCHA.

E. Devanagari Handwritten CAPTCHA dataset:

CSV files are widely used for saving data in a table-like structure, particularly in text editors and spreadsheet programs like Excel. These files contain information that is organized in columns and rows, making it easy to read and manipulate data. Each image file in this project is 65 pixels wide and 65 pixels tall and contains images of every character and numeral. Grayscale versions of these images are extracted using Python code. The results of this project are recorded in a single row of the spreadsheet, with 256 unique values ranging from 0 to FFN. To

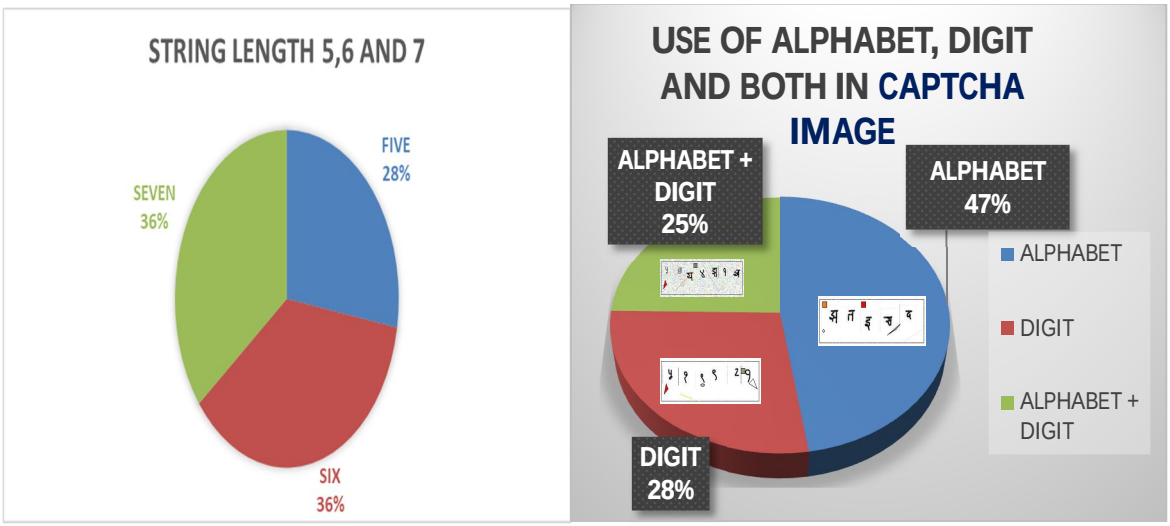


Figure 2. Random CAPTCHA generated of different string length Figure V: Use of Alphabet, Digit, and both in Random CAPTCHA generation

ensure accuracy and compatibility with various tools, each character and digit is saved as a separate CSV file, with one file used for training and the other for testing. The images are also partitioned into separate folders for the training and testing datasets, guaranteeing that the data can be used effectively in various tools and programs. [30]

F. User Interface

The Devnagari CAPTCHA test used a Python-based user interface where users provided their name, age, sex, qualification, and occupation. The test displayed random Devnagari CAPTCHAs of lengths 5, 6, and 7 from three different classes. Users entered symbols using a virtual keyboard with Devanagari characters. The algorithm is implemented using Python to check if the entered symbols match the symbols displayed in the image. If they matched, the user passed; otherwise, they failed.

Approximately 100+ users of different age groups participated in pilot projects at AVISHKAR – District Level (18 Oct. 2022) and AVISHKAR – University Level (11 Nov. 2022) in Jalgaon. All user data was stored in a database. The results showed that 93% of users passed the test successfully on the first attempt, with a 7% failure rate. The 7% who initially failed were able to pass the test successfully on their second attempt, resulting in a 100% success rate for this group on the second attempt, with a 0% failure rate.

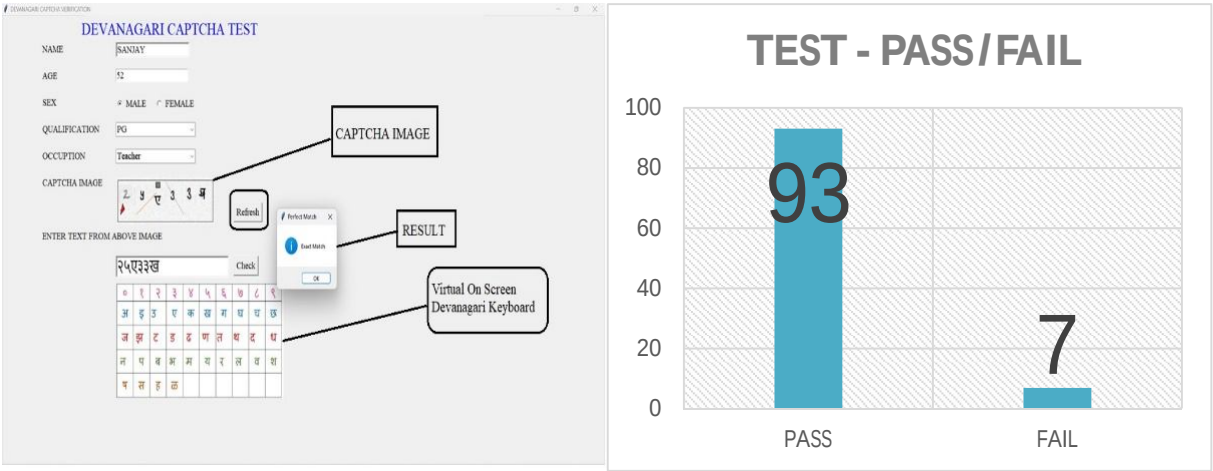


Figure 3. Captcha User Interface

The Devanagari CAPTCHA Test yielded an initial success rate of 93%, with 7% of participants passing on their second attempt.

VI. Conclusion and Future Work

The present study delves into the incorporation of 34 Devanagari handwritten characters and 10 digits in the creation of a robust CAPTCHA system. Our methodology encompasses standard image processing operations, such as image acquisition, resizing, and binarization, as well as advanced techniques like bicubic and bilinear interpolation for scaling and rotation. To enhance the security of the CAPTCHA image against OCR and Decaptcha techniques, we have implemented various types of noises, including Gaussian, localvar, Poisson, salt, pepper, salt & pepper, and speckle.

The importance of accounting for image transformations, such as scaling and rotation, when creating secure CAPTCHA systems was demonstrated through an investigation into their effects on Devanagari symbol images. The results showed that these transformations led to a decrease in both image recognition accuracy and optical character recognition (OCR) performance, highlighting the significance of this consideration for enhancing CAPTCHA security.In the future, we aim to establish a comprehensive evaluation framework for Devanagari CAPTCHA images and assess their recognition and breakage rates. Furthermore, we plan to enhance the complexity of CAPTCHA images by introducing multiple typefaces and fonts, making them more challenging for OCR recognition. Additionally, we envision exploring the possibilities of audio and video Devanagari CAPTCHAs, as well as developing variants without the traditional 'ShiroRekha.' This presents exciting opportunities for further research and innovation in the field of online security. We will also concentrate on enhancing the user experience by refining the virtual keyboard interface and ensuring that the CAPTCHA-solving process is user-friendly. By collecting more feedback from users, we can identify areas of improvement and iterate on the CAPTCHA system to enhance both security and user experience. Moreover, we will optimize the CAPTCHA system for mobile devices, given the increasing use of smartphones and tablets. We plan to implement features that make the CAPTCHAs more accessible for users with disabilities, ensuring inclusivity while maintaining security. Finally, we will consider adopting the Handwritten Devanagari CAPTCHA system for use in a global context, supporting multiple languages and character sets.

Acceptance of Funding

No financial support was given to the authors to create the dataset that is included in the research.

Easy Access to the Dataset

The following is a public dataset of our recently generated Digital Handwritten Devanagari characters:

1. Mendeley (Mendeley Data, “Devanagari Handwritten CAPTCHA - Dataset of 90 K Images: A Challenge Test”, Mendeley Data, DOI: 10.17632/v9wwkvdjmm.1, 30th November 2023.
2. SANJAY E. PATE, PROF.DR.RAKESH J. RAMTEKE, December 11, 2023, "Devanagari Handwritten CAPTCHA - Dataset of 90 K Images: A Challenge Test", IEEE Dataport, doi: https://dx.doi.org/10.21227/w9xw-sd30, 11th December 2023.

References

[1] Devanagari Details of Devanagari script in Wikipedia, https://en.wikipedia.org/wiki/Devanagari,2023. Accessed April, 2024.

[2] Shirali-Shahreza, M.H. &Shirali-Shahreza, M., (2007) “Multilingual CAPTCHA”, ICCC 2007 IEEE International Conference on Computational Cybernetics, 19–21 October.

[3] M. Hassan Shirali-Shahreza, Mohammad Shirali-Shahreza, "Advanced Nastaliq CAPTCHA", Cybernetic Intelligent Systems 2008, IEEE.

[4] Sushma Yalamanchili and Kameswara Rao, “A FRAMEWORK FOR DEVANAGARI SCRIPT- BASED CAPTCHA”, Cryptography and Security (cs.CR); Human-Computer Interaction, International Journal of Advanced Information Technology, Vol. 1, No. 4, August, pp. 47-57, 2011.

[5] https://trends.builtwith.com/widgets/captcha/

[6] T.S.Ravi Kiran, Y. Rama Krishna, "COMBINING CAPTCHA AND GRAPHICAL PASSWORDS FOR USER AUTHENTICATION”, IJRIM, Volume 2, Issue 4 (April 2012) (ISSN 2231-4334).

[7] Bilal Khan, Khaled Alghathbar et.al, “Cyber Security using Arabic CAPTCHA Scheme”, The International Arab Journal Information Technology, Vol. No.1, January 2013.

[8] Baljit Singh Saini, Anju Bala, “Bot Protection using CAPTCHA: Gurmukhi Script”, International Journal of Application or Innovation in Engineering & Management (IJAIEM), May 2013, Volume 2, Issue 5. (ISSN 2319-4847)

[9] M. Tariq Banday, "A Model for Indian Regional Multi-lingual CAPTCHA Challenges",9th J&K Science Congress and Regional Science Congress, 1-3 October 2013, University of Kashmir, Srinagar.

[10] V. K. Yadav, S. Agarwal, J. Uprety and S. Batham, "SRTS: A Novel Technique to Generate Random Text," 2014 International Conference on Computational Intelligence and Communication Networks, 2014, pp. 268-272, DOI: 10.1109/CICN.2014.68.
[11] Sanjay E. Pate," A CAPTCHA: A Review ", National Conference on Recent Trends in Computer Science & Applications 2015. , Date: 27 & 28 January 2015. ISBN No.978-93-84093-76-1(1).
[12] S. A. Alsuhibany and M. T. Parvez, "Secure Arabic Handwritten CAPTCHA Generation Using OCR Operations," 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), 2016, pp. 126-131, DOI: 10.1109/ICFHR.2016.0035.
[13] Seema D. (2017). Digital India Opportunities and Challenges, International Journal of Science & Technology (IJSTM) Special Issue NCIETM – 2017, 6(3): 61- 671
[14] M. Tang, H. Gao, Y. Zhang, Y. Liu, P. Zhang and P. Wang, "Research on Deep Learning Techniques in Breaking Text-Based Captchas and Designing Image-Based Captcha," in IEEE Transactions on Information Forensics and Security, vol. 13, no. 10, pp. 2522-2537, Oct. 2018, DOI: 10.1109/TIFS.2018.2821096.
[15] Charles Boncelet (2005). "Image Noise Models". in Alan C. Bovik. Handbook of Image and Video Processing.
[16] P. Panwar, Monika, P. Kumar, and A. Sharma, "CHGR: Captcha generation using Hand Gesture Recognition," 2018 Conference on Information and Communication Technology (CICT), 2018, pp. 1-6, DOI: 10.1109/INFOCOMTECH.2018.8722409.
[17] Kumar M, Jindal MK and Kumar M (2021), A systematic survey on CAPTCHA recognition: types, creation and breaking techniques. Archives of Computational Methods in Engineering, Springer, 1–30.
[18] Mohinder Kumar1, Manish Kumar Jindal," Benchmarks for Designing a Secure Devanagari CAPTCHA", Published online: 19 January 2021, part of Springer Nature 2021
[19] Asadullah Kehar, Rafaqat Hussain Arain et.al., Design and Development of Sindhi Text-Based CAPTCHAs for Regional Websites, SJET | P-ISSN: 2616-7069 |E-ISSN: 2617-3115 | Vol. 4 No. 1 January – June 2021.
[20] U. Bhattacharya and B. B. Chaudhuri, "Handwritten numeral databases of Indian scripts and multistage recognition of mixed numerals," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 31, no. 3, Mar. 2009, pp. 444–457.
[21] Ashok Kumar Pant, https://www.kaggle.com/datasets/ashokpant/devanagari-character-dataset
[22] Rajiv Kumar, Kiran Kumar Ravulakollu, Handwritten Devnagari Digit Recognition: Benchmarking On New Dataset, Journal of Theoretical and Applied Information Technology 28th February 2014. Vol. 60 No.3
[23] S. Acharya, A.K. Pant and P.K. Gyawali, Deep Learning Based Large Scale Handwritten Devanagari Character Recognition, In Proceedings of the 9th International Conference on Software, Knowledge, Information Management and Applications (SKIMA), pp. 121-126, 2015.
[24] SHALAKA PRASAD DEORE, and ALBERT PRAVIN, "Devanagari Handwritten Character Recognition using fine-tuned Deep Convolutional Neural Network on the trivial dataset", Indian Academy of Science, https://doi.org/10.1007/s12046-020- 01484-1Sadhana(0123456789().,-vols)FT3](012345.
[25] D.S. Prashanth, R.V.K. Mehta, and N.P. Challa," A multi-purpose dataset of Devanagari script comprising of isolated numerals and vowels", https://doi.org/10.1016/j.dib.2021.107723.
[26] Sanjay E.Pate, Prof.Ramteke Rakesh J., "Handwritten Devanagari Character Data Set of 44,000 Images for Devanagari CAPTCHA generation", IEEE data port (DOI: 10.21227/9zpv-3194)
[27] Prof.Ramteke Rakesh J, Sanjay E. Pate, "Handwritten Devanagari Character Data Set of 44,000 Images for Devanagari CAPTCHA generation", Mendeley Data (DOI:: 10.17632/ b9rmfjzc2.1)
[28] Sanjay E. Pate ,Prof. Rakesh J. Ramteke, "DEVANAGARI CAPTCHA DATASET OF 1 Million Images: A challenge Test", IEEE Data Port, DOI: https://dx.doi.org/10.21227/qtmn-m570, 11th April 2023.
[29] Sanjay E. Pate ,Prof. Rakesh J. Ramteke, "DEVANAGARI CAPTCHA DATASET OF 1 Million Images: A challenge Test", Mendeley Data, DOI: 10.17632/knmbfjsdwn.1, 5th April 2023.