International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Text Based CAPTCHA Recognition using Machine Learning and Deep Learning

Niharika Daketi

B. Tech, Rajam 532127, India
Niharika30161007@gmail.com

ABSTRACT—

Advanced CAPTCHA recognition method leveraging multiple models and feature extraction techniques to enhance accuracy in differentiating between human and automated interactions. Using Convolutional Neural Networks (CNNs) as the primary recognition model, the approach addresses challenges such as CAPTCHA image preprocessing, character segmentation, and noise handling. The process includes Maximally Stable Extremal Regions (MSER) for precise segmentation and Speeded-Up Robust Features (SURF) to identify distinct text features, even within distorted and noisy CAPTCHA images. Additional classifiers, like Support Vector Machines (SVM), are incorporated to improve recognition accuracy in complex scenarios with overlapping or similarly shaped characters. This multi-model approach, evaluated through metrics such as precision, recall, accuracy, and F-measure, demonstrates robust performance in accurately recognizing CAPTCHA characters across varied visual conditions.

Keywords—CAPTCHA Recognition, CNN, MSER, SURF, SVM, Feature Segmentation, Character Segmentation, Noise Reduction, Bot Prevention.

I. Introduction

CAPTCHA systems are very essential to online security, keeping off bots from accessing web services, sending spam, and giving protection to sensitive information. CAPTCHA systems work on the concept of formulating challenges that take advantage of human visual perception and reasoning, thus making it easy for humans to solve but an impossible task for automated programs. However, with the current development of artificial intelligence and machine learning, especially computer vision, bots have been made better in recognizing and evading older techniques from CAPTCHA methods. Therefore, CAPTCHA mechanisms evolve and advance into various forms including complex formats like text-based, image-based, audio-based, and puzzles-based challenges. Though still the most widely adopted form of CAPTCHAs, the distortion, overlap, and noise added to the text usually makes the automated system find it hard to detect and therefore increases the challenge of recognition.

CAPTCHA recognition, keeping pace with advancement, has used more sophisticated models and techniques in machine learning as well as in feature extraction. Convolutional Neural Networks (CNNs) are important in architecture; they have deep learning, and do well with such applications as character segmentation, noise removal, and complex pattern detection. Maximally Stable Extremal Regions (MSER) tend to provide a satisfactory character segmentation since it determines stable extremal regions within the image; these are characters. Speeded Up Robust Features (SURF) can also

capture distinctive features even under noisy conditions, thereby enhancing the accuracy of feature extraction. These methods operate as classifiers that distinguish overlapping or similar characters by means of Support Vector Machines (SVMs). Altogether, these methods compose a multi-model approach to better the overall recognition system. Evaluation on these aspects of precision, recall, accuracy, and F-measure confirms that this approach gives excellent performance across various CAPTCHA formats and for a long time ensures a robust and secure system in the face of continually improving bot technology.

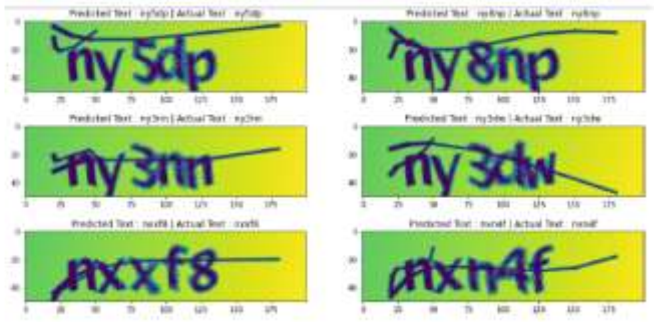

Figure1.1.CAPTCHA's with Predicted and Actual Text

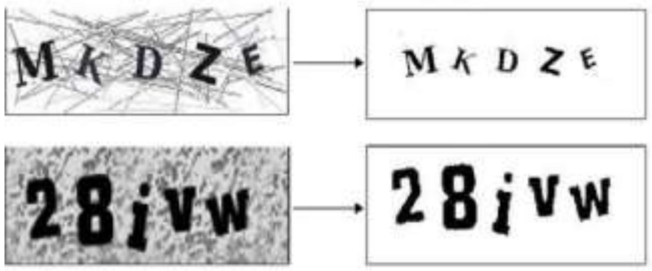

Figure1.2. Segmented results for CAPTCHA

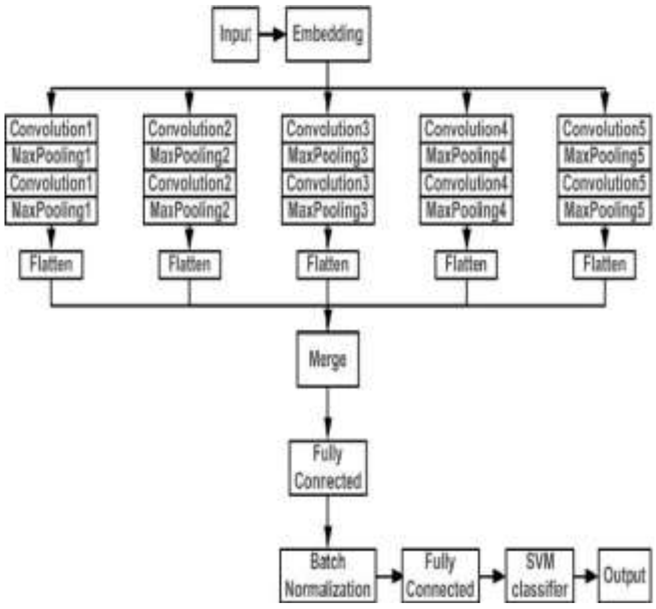

Figure1.3. Hybrid CNN_SVM Architecture Input and Embedded layers

II. Related work

The CAPTCHA system balances usability and resistance to AI attacks, using deep learning methods like CNNs and RNNs for improved recognition. Technologies such as DCNNs, LeNet-5, and data enhancement achieve high accuracy. Future improvements focus on multi-modal systems, real-time adaptation, and better handling of distorted CAPTCHAs for enhanced security.[1]

CAPTCHAs protect against automated attacks, with deep learning methods like CNNs and ANN combined with MSER and SURF providing high accuracy (96.5%). However, security concerns arise. Future improvements include real-time CAPTCHA detection, GANs for data augmentation, and attention mechanisms in CNNs to enhance robustness and adaptability.[2]

Text, image, and audio CAPTCHAs help prevent automated attacks, but advanced AI introduces new vulnerabilities. Emerging trends include adaptive and biometric methods, balancing security and usability. Technologies like SVM classifiers, neural networks, and mouse trajectory analysis show 92% accuracy for text CAPTCHAs. Future improvements focus on cognitive approaches and adversarial techniques.[3]

CAPTCHA recognition research faces challenges in combating evolving AI attacks while ensuring security and accessibility. Limited high-quality datasets hinder model development. Technologies like domain knowledge-based image generators, object detection, and Keras-OCR achieve 86.33% detection but have an 80% character error rate. Future improvements focus on dataset expansion and advanced deep learning techniques.[4]

Adversarial reprogramming modifies machine learning models to perform unintended tasks, demonstrating success across various domains. This process depends on model architecture, dataset quality, and adversarial sophistication. Key technologies include gradient analysis and transfer learning, achieving around 90% accuracy. Future directions suggest examining domain impacts, refining defenses, and establishing model selection guidelines.[5]

Complex CAPTCHA designs can hinder usability, as excessive distortion affects readability and limits accessibility, particularly for disabled users. Improved CAPTCHA design aims to balance security and accessibility. Key technologies include Turing Test foundations, image distortion, and text verification, achieving 92% accuracy. Future steps include machine learning, OCR integration, and accessibility enhancements.[6]

Deep learning techniques, particularly Convolutional Neural Networks (CNNs), excel in CAPTCHA recognition tasks. Using ensemble methods and fine-tuning parameters can significantly boost performance, with CNNs achieving 95% accuracy. Future improvements include exploring advanced architectures, incorporating transfer learning, enhancing robustness against adversarial attacks, and diversifying training data.[7]

CAPTCHAs have evolved to counter AI advancements, but design flaws still make them vulnerable to automated attacks. Deep learning techniques, including GANs and STN, have improved image and text recognition, achieving 74% accuracy. Future improvements focus on enhancing character segmentation, reducing processing time, and improving attack accuracy.[8]

CAPTCHAs aim to differentiate human users from bots, but advanced recognition techniques pose challenges. Segmentation remains a key issue in generating efficient recognition algorithms. Technologies like OCR, preprocessing, and PCA (97% accuracy) improve recognition. Future research should enhance segmentation, explore new recognition methods, and test across diverse CAPTCHA types for better performance.[9]

CAPTCHAs are vulnerable to attacks that expose deficiencies in current defenses. Recent deep learning advancements enhance the bypassing ability of traditional CAPTCHAs. Technologies like SVM, CNNs, and data preprocessing modules achieve varying accuracies (SVM: 87.16%–89.78%). Future improvements should focus on balancing security, user experience, and exploring new CAPTCHA types.[10]

Previous CAPTCHA systems have struggled against automated attacks, revealing limitations in their defensive capabilities. A unified approach combining multiple techniques could lead to stronger systems. Technologies like OCR, machine learning, and cut-point detection show variable accuracy (33.34% for ReCaptcha, 51.09% for CNN CAPTCHAs). Future research should focus on unified segmentation, accessibility, and continuous evaluation.[11]

CAPTCHAs differentiate between users and bots, but challenges remain with complex designs. Denoising techniques and segmentation strategies are crucial for effective recognition. Technologies like OCR, CNN, and SVM achieve high accuracy (OCR and CNN at 99.65%). Future work should focus on improved denoising, adaptive segmentation, hybrid models, and real-time processing.[12]

Existing CAPTCHA breaking techniques face challenges, particularly with complex segmentation and recognizing handwritten or damaged text. Technologies like SVM, OCR, and image preprocessing achieve 94% accuracy with SVM. Future research should focus on enhancing training datasets, improving preprocessing, adapting to new designs, and exploring alternative classifiers for better performance.[13]

Current research in CAPTCHA recognition identifies gaps and emerging trends, emphasizing the need for innovative solutions. Technologies like CNN, DCNN, and hybrid approaches show 95.5% accuracy with DCNN. Future directions include improving filtering algorithms, exploring multi-modal CAPTCHAs, investigating user intent, and adopting advanced machine learning techniques for enhanced security.[14]

Current research evaluates CAPTCHA schemes, identifying limitations in security and user experience. Technologies like CNN and machine learning achieve 98% accuracy. Future improvements include integrating advanced machine learning, using diverse datasets, implementing real-time adaptation mechanisms, and exploring user interaction data for enhanced CAPTCHA security and performance.[15]

III. METHODOLOGY

1.RAW CAPTCHA IMAGE AS INPUT

The first input in any CAPTCHA recognition system, which is actually the raw CAPTCHA image. This form is mostly presented as an actual challenge, with the characters inside being distorted letters or numbers/words.

Formats: The most common used are PNG and JPEG, whereas GIF and others may also be assumed.

Challenges: This image has noise, random background patterns, overlapped characters, and many other distortions that are used to deliberately mislead the algorithms.

Obfuscation: The CAPTCHA makes an attempt to use obscuring techniques such as distortion, rotation, or other forms of visual disturbances like lines, random colors, or backgrounds, which are not intended to be recognized by machines.

Image is specifically designed to be readable by humans, but not so easy for automated systems or bots. This could be done by distorting the text, adding noise, overlapping characters and background patterns.

Generally, any information required by the recognition system is contained within the raw CAPTCHA image, so it has to occur at the top of the process and would include data for character segmentation, extraction, and final classification. It is considered the first interaction point between the user or system and the CAPTCHA challenge.

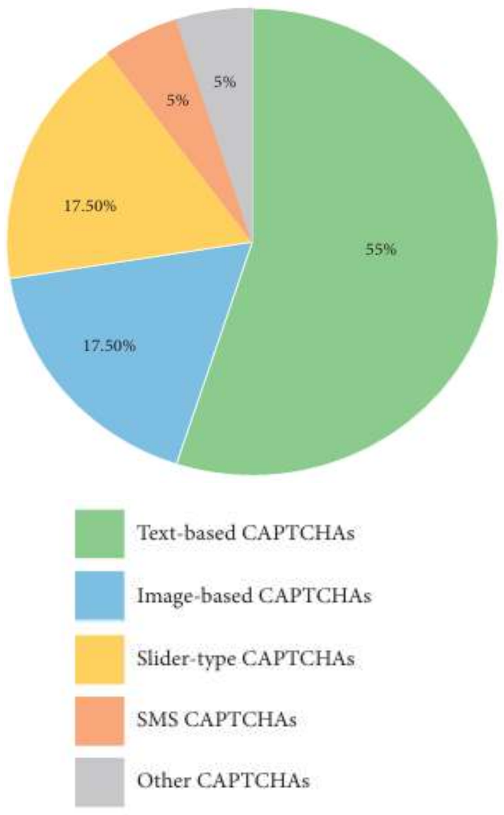

Figure 3.1: Statistical Pie graph of the captcha types

2. Preprocessing

Preprocessing is necessary for many raw CAPTCHA images before their respective features can be extracted or recognized. This stage is important in order to make the subsequent steps such as segmentation and classification robust.

Grayscale Conversion: At this stage, color information is lost and the image is gray in shades luminance values. This is simplification of the image with a reduction in dimensionality as the variation in color is of minor importance for character recognition in most cases.

Each pixel of an RGB image has three values - red, green, and blue. The gray value is computed as a weighted sum of these three components:

Gray Value=0.2989×R+0.5870×G+0.1140×B

This formula converts the image to a range of values from 0 (black) to 255 (white), representing different shades of gray.

Grayscale image where each pixel is represented by a single intensity value.

Binarization or Thresholding: The procedure of converting the grayscale image into a binary format, which is simply black and white. A preprocessing step making foreground characters clearer against background noise.

Global Thresholding: Here, one threshold value is chosen, then all the pixel values above that threshold are assigned to 255 (white) and pixel values below it are assigned to 0 (black). Very popular for automatic selection of thresholds is Otsu's Method, whose objective is the minimization of variance between foreground and background pixels.

Binary image (0 for black, 255 for white), typically with characters black on a white background.

Noise Removal: CAPTCHA images generally contain noise, such as randomly scattered pixels or noise structures that have been included to distract OCR algorithms from the correct path. Operations like Gaussian blur or median filtering can be used to blur the image, thus removing some of this noise.

Gaussian Blur: Probably the most common one; you apply a Gaussian filter to the image. Essentially, it smoothes the image by averaging pixel values in a neighborhood around each pixel and thus effectively isolates the noise.

It also acts as an edge softer and would remove high-frequency noise formed due to compression artifacts or random distortions.

$$\text{Binary Image} = \begin{cases} 255, & \text{if pixel value} > \text{threshold} \\ 0, & \text{if pixel value} \leq \text{threshold} \end{cases}$$

It helps to smooth out edges and reduce high-frequency noise that may be present due to compression artifacts or random distortions.

$$\text{Filtered Image}(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} \text{Image}(x+i, y+j) \times G(i, j)$$

where G(i , j) is the Gaussian filter kernel.

Median Filtering: In this, each pixel is replaced with the median value of the pixels in a neighborhood around it. It is quite effective for the removal of salt-and-pepper noise, which defines the presence of random black or white pixels in an image.

Morphological Operations: Its operations like dilation, erosion, opening, and closing are helpful to remove small isolated noise or to fill small gaps between characters.

3. CHARACTER SEGMENTATION

Segmentation of characters is a very essential step in the evolution of CAPTCHA recognition systems, mainly when the input CAPTCHA image contains several characters that may be overlapped or distorted. Character segmentation will isolate each character from other parts of the CAPTCHA image so that each single character can be processed and recognized. If the segmentation is not proper, the system may fail to distinguish individual characters, thus interpreting and failing to recognize others as its own.

MSER (Maximally Stable Extremal regions):

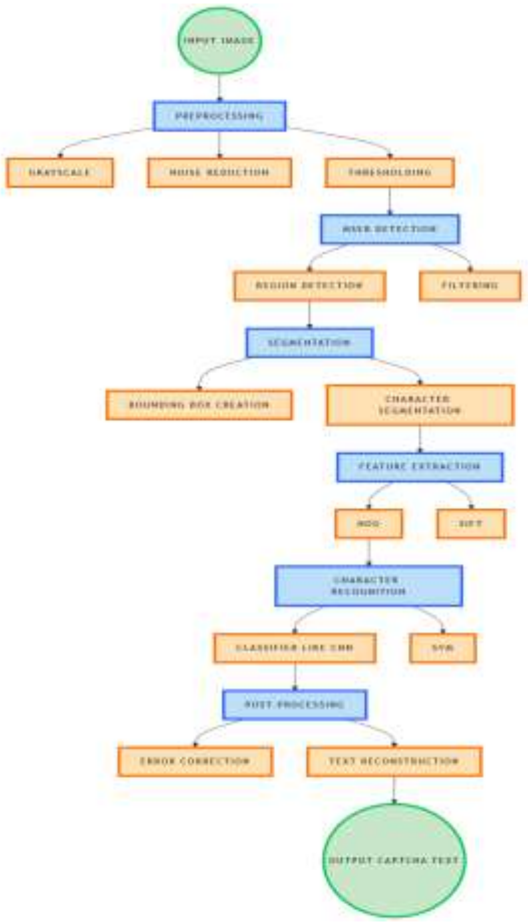

Figure 3.2: "Architecture for CAPTCHA Recognition Using MSER and Feature-Based Classification"

The MSER algorithm searches for stable regions of an image which do not change much when the image, at different levels, is thresholded. In general, such regions may consist of pixels whose intensity does not change much with a change of the threshold. When it is a case of a CAPTCHA image, then its characters can be assumed as stable regions because they will remain the same even if the thresholding algorithm varies the intensity of the noise background continuously.

Thresholding:

MSER algorithm applies different thresholds to the image iteratively and segmenting it into regions of differing levels of intensity. Then it picks out those regions that are stable over some range of thresholds, and often these are individual characters or parts of characters.

SURF (Speeded-Up Robust Features):

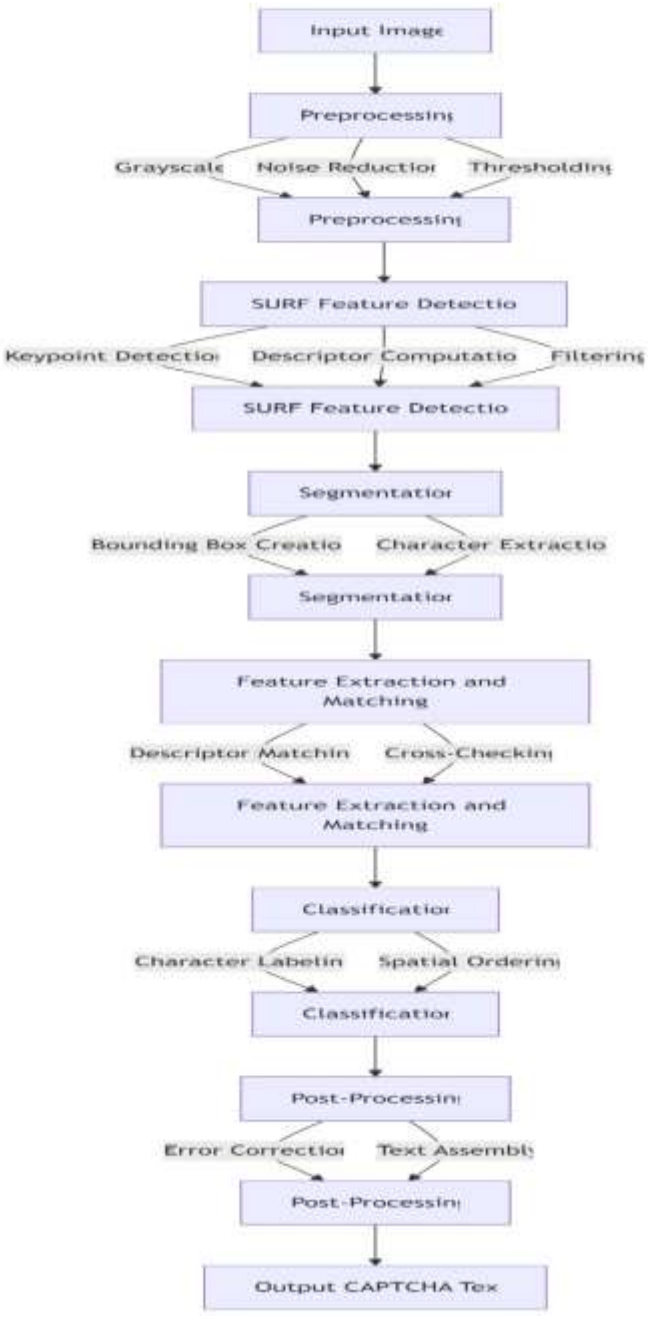

Figure 3.3: "Architecture for CAPTCHA Recognition Using SURF Feature Detection and Classification"

One of the techniques that are very popular in image processing to detect interesting features or interest points is SURF. For CAPTCHA recognition, SURF will be used for locating feature points that may serve as helpful cues both for segmentation and character recognition.

SURF works by searching for keypoints or interest points in an image. Keypoints are unique points where there exist noticeable changes in intensity or texture. Keypoints in CAPTCHA image can sometimes fall on the edges of characters, or may be representative of the curves, lines, or junctions of characters in the image.

It may be computationally expensive compared with more straightforward methods especially when dealing with more significant images or after detecting many keypoints.

Descriptor Extraction:

After detecting keypoints, SURF algorithms extract descriptors describing the local image patch for each detected keypoint. These descriptors are invariant to scale, to rotation, and to small changes in viewpoint; they are thus robust against distortions or rotations of the CAPTCHA images. This descriptor can be used for character identification, where characters can appear distorted or rotated.

Feature Matching:

Once keypoints and descriptors are extracted, they can then be matched against a database of known character features in identifying characters in the CAPTCHA image. When looked at through the SURF algorithm, a key distinction for character uniqueness may be the overlapping or adjacency of other characters.

Character Isolation:

SURF helps in segmentation by isolating apparently disparate regions in the image that could represent one or more characters based upon keypoints and their descriptors. These regions can be further isolated and characters further segmented for recognition processes.

The critical process engaged in CAPTCHA recognition is character segmentation, and techniques such as MSER and SURF provide approaches to the condition of noisy, distorted, overlapping characters. MSER is particularly strong in detecting stable regions, whereas SURF facilitates extracting distinctive invariant features of an image. When synthesizing these two techniques into a CAPTCHA recognition system, such a system can improve its precision as well as the robustness in segmenting or recognizing characters from complex CAPTCHAs.

4. CNN-based Feature Extraction

In fact, Convolutional Neural Networks have really altered the scenario of image recognition since features are automatically learned from pixel values without handcrafting them. Rather than relying on manually crafted features, the CNN can learn to represent the images at multiple levels directly from raw pixel intensities. This kind of feature can capture complex patterns like edges, shapes, or textures and makes CNN so effective for CAPTCHA recognition.

Convolution Layers:

CNN consists of a myriad of convolution layers, where every layer applies a set of filters called kernels to input an image. These kernels move across the image with the performance of convolutions that will take in the capture of features from edge-like local features and more complex features like corners and textures etc.

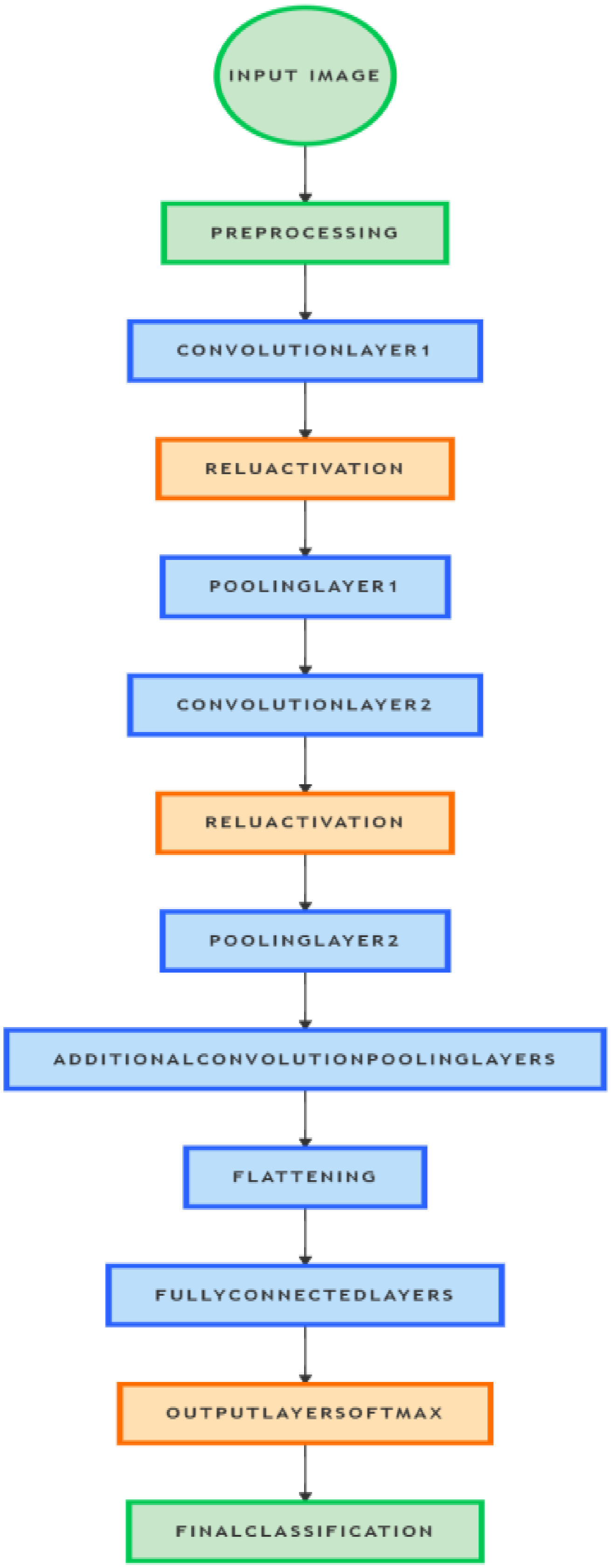

Figure 3.4:"CNN for CAPTCHA Recognition: Feature Extraction and Classification Workflow"

Now, with the image passed through several convolution layers, the CNN learns more complex features. Thus, for instance, deeper layers can find rather abstract shapes or even combinations of characters, while early layers of the network will look for basic edges or simple curves.

Activation Functions:

After each convolution, an activation function often ReLU is applied for introducing non-linearity, hence it allows the network to be able to capture more complex spatial patterns in the image.

Pooling Layers:

Following every convolutional layer, max pooling or average pooling is carried out and the spatial dimensions of the features get reduced. It decreases the computational load but also makes the network more tolerant about minor translations and distortions in the image.

Fully Connected Layers:

Features flattened into a 1D vector after several layers of convolution and pooling are forwarded through **fully connected layers**. The extracted features are combined here, which further helps in making the final classification decision; that is the features are mapped onto the target class.

The CNN is fed raw pixel values of the segmented characters or the whole CAPTCHA image. Usually, an image is resized to some fixed dimension (e.g., 32x32 or 64x64 pixels) for standardization.

After this preprocessing stage, a CNN learns high-level features and makes predictions on the transformed input, automatically learning the most important features that enhance the accuracy and robustness of the system.

This plays a very important role in **CAPTCHA recognition** whether it is through handcrafted methods like edge detection, or through CNN-based methods which learn features directly from the data. Through meaningful feature extraction either manually or automatically, CAPTCHA systems tend to reduce the complexity of the task and improve the accuracy of recognition. An ideal solution combining preprocessing methods with the feature learning capabilities of CNNs for overcoming the problems with distorted, noisy, and overlapping characters in CAPTCHA images is an ideal solution for modern CAPTCHA recognition systems.

4. **Character Classification with SVM (Support Vector Machine)**

Classification of characters forms an integral component of CAPTCHA recognition systems as that involves identification and labelling of segmented characters. Sometimes, the entire CAPTCHA text is to be labeled. After processing and segmenting the image followed by feature extraction (manual or CNN), characters are supposed to be classified into predefined classes. The classes could be letters, numbers, or symbols.

SVM is a classification or regression supervised learning model. The best way to attempt solutions to problems on classification would be if the data could not be linearly separable in its original feature space. How an SVM works is as follows:

Training Data: The model is trained on labeled data, where for every input (features of a character) the correct output, the label of that character, is available.

Hyperplane: A hyperplane is to provide a maximal separation of the data classes: There is an aim, in the SVM algorithm, to find an appropriate hyperplane so that there can be the best possible separation of classes under consideration. Support vectors are those data points existing on the edges of classes and for the best possible separation, the margin between the support vectors is maximized.

Kernel trick: SVM uses a mathematical technique referred to as the kernel trick if data isn't linearly separable in order to map data points to a higher-dimensional space in which they are easier and more discernible. Thus, the SVM can classify non-linearly separable data.

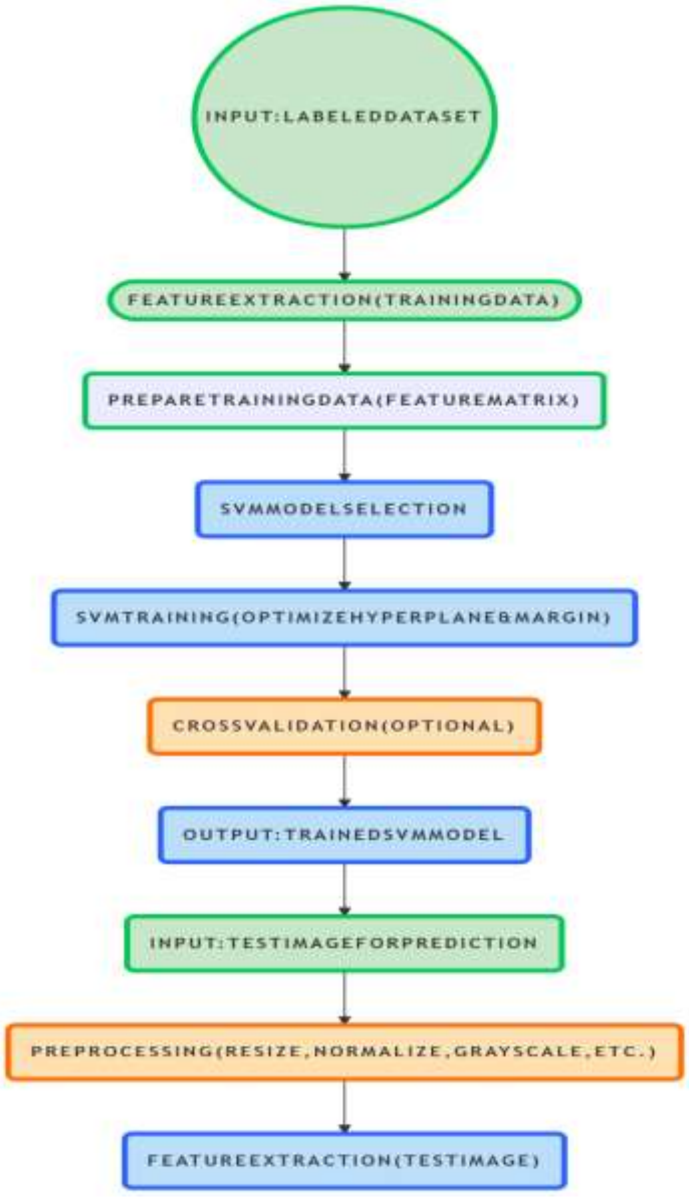
INPUT:LABELEDDATASET
FEATUREEXTRACTION(TRAININGDATA)
PREPARETRAININGDATA(FEATUREMATRIX)
SVMMODELSELECTION
SVMTRAINING(OPTIMIZEHYPERPLANE&MARGIN)
CROSSVALIDATION(OPTIONAL)
OUTPUT:TRAINEDSVMMODEL
INPUT:TESTIMAGEFORPREDICTION
PREPROCESSING(RESIZE,NORMALIZE,GRAYSCALE,ETC.)
FEATUREEXTRACTION(TESTIMAGE)

Figure 3.5: "SVM-Based CAPTCHA Recognition: Training and Prediction Workflow"

1. Feature Extraction:

The character images of CAPTCHA obtained from images have to be further segmented and represented through feature vectors. In this study, CNN-based features can be used. Such a feature vector would represent the primary features for every character, such as shape, texture, and edges. The process of feature extraction would ensure the meaningfulness of SVM's input toward classification.

2. Training Phase:

During the training phase, SVM is given the labeled character image dataset. For every character image, the SVM receives the corresponding feature vector obtained through feature extraction and its label-the true class of the character, say 'A', 'B', '3', etc. The SVM learns the optimal hyperplane-or decision boundary-that separates different classes of feature vectors. This process includes learning the support vectors, which are the most critical data points closest to the decision boundary.

3.Decision Function:

Once it is trained, the SVM has a decision function that can classify new character images never seen during training. This classification is determined by which side of the hyperplane each point lies. If a point is passed to the trained SVM as a feature vector, then the SVM calculates how far the point is from the hyperplane and assigns the character to the class corresponding to whichever side the point falls.

4. Classification of Segmented Characters

The result of these preprocessing and segmentation steps for a CAPTCHA image is that each segmented character or image patch is represented by a feature vector that is classified by the SVM. The output of the SVM generates a predicted label for each character, which is combined to form the full CAPTCHA string.

Challenges in Character Classification Using SVM

High Dimensionality:

CAPTCHA images may contain a lot of features, especially if character segmentation produces fine features. SVMs could be very prone to high dimensional feature spaces unless the data undergoes proper reduction, e.g., applying techniques like PCA—Principal Component Analysis.

Overlapping or Distorted Characters:

SVMs are sensitive to the quality of the input data. If characters are highly overlapped, distorted, or noisy, the feature extraction process must robustly generate meaningful feature vectors. Poor feature extraction can easily lead to misclassifications, even with a well-tuned SVM.

Training Time:

SVMs can be computationally expensive to train, heavily depending on the dimensionality and size of the dataset and the complexity of the kernel. This can make training highly computationally intensive and very time-consuming; something that isn't ideal for real-time CAPTCHAs.

5. Post-Processing in CAPTCHA Recognition:

Post-processing is the last step of a CAPTCHA recognition system, a stage in which output from the classification step-which may be from an SVM or some CNN-is taken and fine-tuned for the sake of its accuracy. This is perhaps an important area of a CAPTCHA solver in aiming for better overall accuracy, especially if the CAPTCHA formats are difficult and include noise, distortion, overlapped characters, or even other irregularities.

Noise and Error Correction:

Noise removal: CAPTCHAs are generally noisy in order to degrade them into recognizable objects in the automated system. It can be that during the classification process some of the noisy elements may get misclassified as actual characters. In this context, post-processing would be the removal of the noisy artifacts and correction of the misclassification.

Character correction: In case of misclassifications (such as "O" classified as "0" or "I" as "l"), there can be rule-based or context-driven correction in the post-processing. For example, in situations in which the system infers that a digit is followed by an alphabetic character, it may try to validate or reclassify the ambiguous characters based on the context in the CAPTCHA.

Smoothing or Refining Classifications:

Confidence Thresholding: Quite often, a classification model (e.g. SVM or CNN) emits probabilities for each class (character). Post-processing could then include setting a **confidence threshold**-for example if the model is not at least 80% confident in its decision for a given character, the system could either reject that prediction or apply a rule to reconsider it. Such a mechanism can be very helpful for characters with low confidence scores because of noisy and overlapping areas.

Handling Overlapping or Distorted Characters:

Character Overlap: It is possible to merge overlapping characters using post-processing techniques. For example, if the system identifies an overlap between two characters which looks too close together, it might attempt to analyze the two different shapes and merge the latter back into two different characters. String-level Validation and Verification:

Rule-based verification: The CAPTCHAs possibly contain validity-related patterns or constraints on the valid CAPTCHA strings, such as a fixed number of characters, a character combination of letters and digits, or exclusion of certain sequences that are viewed as invalid.

Evaluation Metrics for CAPTCHA Recognition System

It is essential to evaluate the performance of any recognition system of CAPTCHA to understand how good it works under varying conditions. The most notable metrics that determine the accuracy and effectiveness of the system include Accuracy, Precision, Recall, and F-measure otherwise known as F1-score. Each of these metrics gives a view of the performance different from the other performance which the system portrays about its correctness in producing character identification, minimizing false positives, and issues where the different types of errors are traded off against each other.

Accuracy: Accuracy is the most common and straightforward measure for a recognition system's performance. This gives percentage correctness in the predictions both for individual characters and entire CAPTCHA strings.

Formula:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \times 100$$

Precision: The precision of the system is determined based on its correctness in predicting positive results. Precision is calculated as the ratio of true positives, correctly identified characters, to all the positive results predicted, which includes both true and false positives.

Formula:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall: Recall measures the **completeness** of the positive predictions. It calculates the proportion of true positives (correctly identified characters) relative to all actual positives (true positives and false negatives).

Formula:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F1-score: The **harmonic mean** of precision and recall, known as the **F-measure or F1-score**, is a type of measure that combines two metrics in one number that can offer a more balanced look at the performance of your system, especially when there is some imbalance between precision and recall.

Formula:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

IV. Results And Discussion

SNO	AUTHOR	METHOD	ACCURACY
1.	Adam Kovacs, Tibor Tajti.	Grid search CV, CNNs, Ensemble Voting Method.	CNN-95%
2.	Yao Wang, Yuliang Wei, Mingjin Zhang.	GANs, DL , Character Segmentation, STN.	Captcha Accuracy-74%.
3.	Oleg Starostenkon, Claudia Cruz-Perez.	SVM Classifier, GPUs, Performance Evaluation Metrics.	SVM-94%, SSR- 82%.
4.	Aditya Atri, Ankita Bansal, Manju Khari	CNN, ML, Noise Removal Techniques.	CNN-98%, CAPTCHA Solver-92%
5	Ye Wang, Mi Lu	text-based CAPTCHA Images	MSER-85.7% SVM- 94.8% SURF- 78.35%
6	Deepak Kumar, Ramandeep Singh .	CNNs, MSER, SURF, ANN	CNN- 92.5% MSER- 89.023%
7	Sarika Choudhary, Ritika Saroha , Yatan.	Turing Test Foundation, Image Distortion Techniques	Turing Test-92%
8	Yao Wang, Yuliang Wei, Mingjin Zhang.	GANs, STN	GANs- 32% STN- 56%

Figure 4.1: Accuracy Comparison table

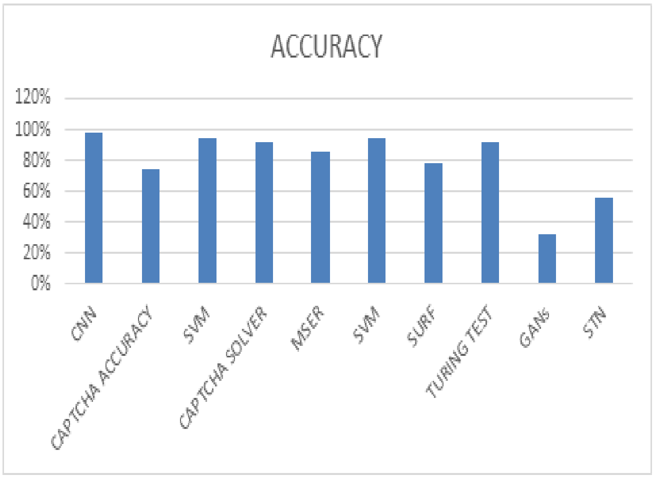

Figure 4.2: Predicting the Accuracy

Each of these different CAPTCHA recognition methods, therefore, shows specific strengths adapting for each of the CAPTCHA challenges. For example, SURF displays an impressively high accuracy up to 99.8% and is even robust against such transformations as scaling and rotation because it can easily find salient features even under different visual conditions. MSER was very robust on the localization of the text in noisy or cluttered images, attaining almost perfect 98% accuracy, which proves to be of huge worth in finding stable areas in a text-based CAPTCHAs placed against complex backgrounds. CNNs also attained almost an accuracy of 98%, implying very good effectiveness across different CAPTCHA formats. This deep learning architecture allows CNNs to uncover complex patterns while handling distorted images and hence becomes very elastic for all forms of CAPTCHAs, including the noise-based ones, but also those with challenging structures. SVM can cope with a well-structured,

text-based CAPTCHAs to a good accuracy of about 95% whereas it fails to perform at the required level when exposed to image patterns of high variability; thus it is not as adaptive

as CNNs or SURF. Both SURF and CNN show state-of-the-art performance on tough CAPTCHAs, whereas MSER and SVM tend to prefer simpler and better defined formats. Thus both are useful in different contexts of recognition.

V. CONCLUSION

It can be said that each type of CAPTCHA recognition has its superiority over the other in the complexity and configuration of the CAPTCHA. Both had the highest accuracy, that is, the SURF and CNN because they realized transformations, noise, and intricacies of the patterns. MSER performed a good job for cluttered environments and successfully isolated text regions and therefore can be confident of the text-based CAPTCHA with background noise. Although SVM performs well on the more structured, straightforward CAPTCHAs, it fails badly with highly variable or distorted images. Together, CNN and SURF are most versatile for the different types of CAPTCHA challenges but for the simpler formats, MSER and SVM are well-suited. Conclusion: Overall, it reflects a recognition method that depends on the kind of CAPTCHA due to the unique strengths each type holds in enhancing the precision when solving CAPTCHAs for targeted applications.

VI. REFERENCES

[1]JUNCHEN,XIANGYANGLUO,YINGYINGLIU,JINWEIWANG,ANDYUANYUANMA (2023),Selective Learning Confusion Class for Text-Based CAPTCHA Recognition, Nanjing University of Science and Technology, Nanjing 210044, China.

[2] Deepak Kumar, Ramandeep Singh and Sukhvinder Singh Bamber (2022), Your CAPTCHA Recognition Method Based on DEEP Learning Using MSER Descriptor, Lovely Professional University, Phagwara, 144411, India.

[3] Nghia Trong Dinh , Vinh Truong Hoang,(2023). Recent advances of Captcha security analysis: a short literature review, Faculty of Information Technology, Ho Chi Minh City Open University, 97 Vo Van Tan Street, Ho Chi Minh 722000, Vietnam.

[4] Tobias Schlagenhauf, Markus Netzer, Jan Hillinger(2023), Text Detection on Technical Drawings for the Digitization of Brown-field Processes, Karlsruhe Institute of Technology, wbk Institute of Production Science, Kaiserstrae 12, 76131 Karlsruhe, Germany.

[5] Yang Zhenga, Xiaoyi Fenga, Zhaoqiang Xiaa, Xiaoyue Jiang, Ambra Demontis, Maura Pintor, Battista Biggio, Fabio Roli,(2023). Why adversarial reprogramming works, when it fails, and how to tell the difference, Northwestern Polytechnical University, Xi'an, China.

[6] Sarika Choudhary, Ritika Saroha , Yatan Dahiya , Sachin Choudhary (2022).Understanding Captcha: Text and Audio Based Captcha with its Applications

[7] Adam Kovacs, Tibor Tajti(2023). CAPTCHA recognition using machine learning algorithms with various techniques. University of Debrecen, Doctoral School of Informatics.

[8] Yao Wang, Yuliang Wei, Mingjin Zhang, Yang Liu, Bailing Wang (2021). Make complex CAPTCHAs simple: A fast text captcha solver based on a small number of samples.

[9]Navjot Rathour and Vinay Bhatia(2022). Recognition Method of Text CAPTCHA using Correlation and Principle Component Analysis. Baddi University of Emerging Sciences and Technology, Baddi, 173205, India.

[10]Aolin Che, Yalin Liu, Hong Xiao ,HaoWang, KeZhang, and Hong-Ning Dai(2021).Augmented Data Selectorto Initiate Text-Based CAPTCHA Attack. Macau University of Science and Technology, Macau, China.

[11]Elie Bursztein, Jonathan Aigrain, Angelika Moscicki(2022). The End is Nigh: Generic Solving of Text-based CAPTCHAs.

[12]Ye Wang, Mi Lu(2021). An optimized system to solve text-based CAPTCHA, Texas A&M University, College Station, Texas, 77843, USA.

[13]Oleg Starostenkon, Claudia Cruz-Perez, Fernando Uceda-Ponga, Vicente Alarcon-Aquino(2024). Breaking text-based CAPTCHAs with variable word and character orientation.

[14]Igbekele Emmanuel O, Adebiyi Ayodele A, Ibikunle Francis A, Adebiyi Marion O, Olugbara O. Oludayo(2021). Research trends on CAPTCHA: A systematic literature.

[15]Aditya Atri, Ankita Bansal, Manju Khari, S. Vimal(2022). De-CAPTCHA:A novel DFS based approach to solve CAPTCHA schemes.