International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 09 | Sep 2022 www.irjet.net p-ISSN: 2395-0072

# Demonstrated Deep Learning Techniques for the Resolution of CAPTCHA images

**Bhavya Shah[1], Kunal Kulkarni[1]**

[1]*Student, Department of Analytics, Vellore Institute of Technology, Tamil Nadu, India*

---

**Abstract -** *The research conducted has the aim to deconstruct text-based CAPTCHAs via the aid of Deep Learning methods. Our objective is to train a neural network on various CAPTCHA datasets that can guess the output of the CAPTCHA image we wish to break after image processing is applied to the image to extract the relevant information from it. The motivation for this project stemmed from our desire to develop an effective CAPTCHA solver that can break text-based CAPTCHAs of any design without relying on hard-coded rules and constraints. We also wish to expose the underlying weaknesses in the security mechanism of text-based CAPTCHAs. We intend to train a neural network model. The second part of our project focuses on the creation of a Python script that can put our model to use to solve CAPTCHA images in real-time. The developed CAPTCHA solver should be able to break any text-based CAPTCHA image fed to it and provide the correct output. The neural network should not be prone to overfitting.*

***Key Words***: Deep Learning, CAPTCHA Detection, Image Processing, Analytical Reasoning

## 1. INTRODUCTION

Deep learning is a machine learning technique based on the idea that computers can learn by example in the same way humans do. In this technique, a computer model learns to execute classification tasks from text, images and audio directly by training the model on a vast set containing labelled data and neural network architectures comprising multiple layers. It is ideally suited for predicting outcomes when there is a large amount of data to learn from so that the system possesses enough data to train itself. One of the key benefits of deep learning is its ability to solve complex issues that necessitate the discovery of hidden patterns in data and/or a thorough knowledge of complex relationships among a large number of interdependent variables. Deep learning algorithms can discover hidden patterns in data on their own, combine them, and create far more efficient decision rules. Deep learning excels in complicated tasks that often require dealing with large amounts of unstructured data, such as picture classification, natural language processing, and speech recognition, to name a few. The task of solving CAPTCHA images can be considered to be a part of this problem domain. Hence, we have chosen to use deep learning to solve CAPTCHAs.

In this study, we intend to develop a Deep Learning model that can successfully break any text- based CAPTCHA image by identifying the patterns and structure in the CAPTCHA image. Since the Deep Learning method requires a vast dataset to train the model, we will use a large labelled dataset consisting of CAPTCHA images. Data pre-processing will need to be applied to extract relevant features from it. The processed data will serve as the training and test datasets. With the help of pre- trained models, our Deep Learning model will be trained on the training data and subsequently evaluated using important metrics like accuracy on the test data.

The main objective of this study is to develop a model that can identify the pattern in the CAPTCHA image. The methodology starts with the pre-processing of an image dataset that will be used to train our model. Pre- processing involves steps like separating and encoding characters from the target, reshaping and resizing images. The next step is to create a data pipeline that will help in feeding images to our model. Then, the architecture of our model is created. We use pre-trained models like VGG16, MobileNet and DenseNet for the model. The penultimate step is to train the model on our training dataset. The final step is to test its accuracy and various other metrics on the test dataset.

### 1.1 The Roles and Significance of CAPTCHA: An Overview

The CAPTCHA stands for Computer Automated Public Turing Test to Tell Computers and Humans Apart. This security measure falls under the category of challenge-response authentication. It offers protection against spam and password decryption by requiring the user to complete a test that is simple for humans but not for computers, thus helping to verify whether it is a human or a computer attempting to access a password-protected account.

A CAPTCHA test consists of a distorted image comprising a randomly generated sequence of letters and/or digits and a textbox placed below it. The user needs to just type the characters discerned in the CAPTCHA image into the textbox so that the user's human identity can be confirmed and consequentially, the test is passed. For visually impaired people, an audio version of the CAPTCHA image is available. The principle behind the functioning of a CAPTCHA is that computers can develop distorted images and process the response provided but they do not have the reading and solving capabilities of humans required to pass the CAPTCHA test.

CAPTCHA tests are based on unsolved problems in Artificial Intelligence (AIHence, there are two positive sides to this situation of CAPTCHAs. The first possibility is that the CAPTCHAs remain unbroken so the objective of being able to differentiate between humans and computers is fulfilled. The other possibility is that the CAPTCHA is broken leading to an AI problem getting solved. Thus, our project's aim depends on this premise.

### 1.2 Existing Systems and Statistics: A detailed discussion

There exist many current popular methods to solve CAPTCHAs. These include, but are not limited to the ones listed below.

Optical Character Recognition (OCR) enabled bots: Tools such as Ocrad and tesseract solve CAPTCHAs automatically using the Optical Character Recognition technique in which text is detected and read in images using computer vision.

Machine learning: Deep convolutional neural network models can be trained to detect characters in a CAPTCHA image. Technologies including computer vision, convolutional neural networks and python frameworks/libraries like Keras and TensorFlow are utilised in this method.

Online CAPTCHA-solving services: Human workers are available online 24 hours a day, 7 days a week to solve CAPTCHAs. When a CAPTCHA solving request is submitted, the service passes it on to the solvers, who decipher it and return the answers.

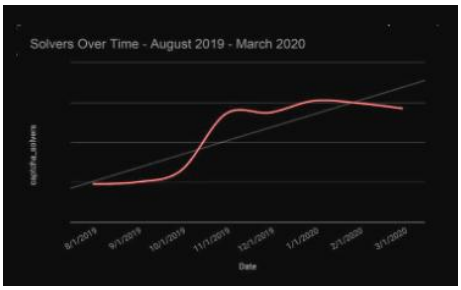


Figure 1. Increase in popularity of CAPTCHA solvers due to their accessibility and ease of use.

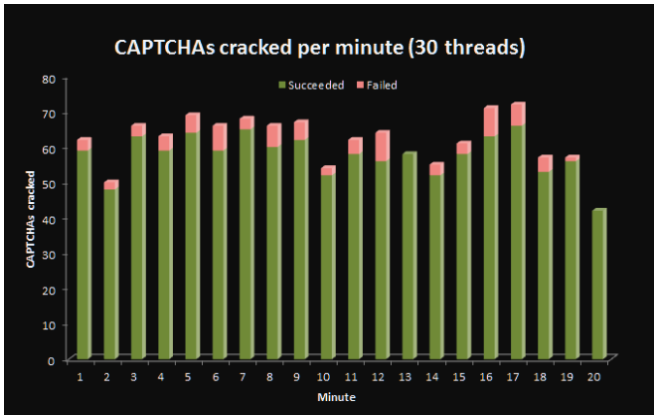


Figure 2. Graph of CAPTCHAs cracked per minute by an online CAPTCHA-solving service called Antigate (94% accuracy achieved).

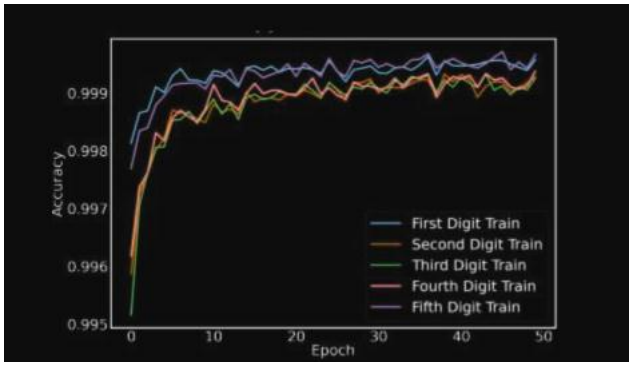


Figure 3. Accuracy metric of the Adam optimizer in ML on a dataset of 500

### 1.3 Benefits and Challenges of Proposed Method

The use of automated CAPTCHA solving is permissible as well as prohibited. Because they are compensated based on how much spam they produce, spammers have an incentive to collect as many email addresses as they can, and CAPTCHA stands in their way. They therefore require a low-cost method of getting around the CAPTCHA security. When a party attempts to "skew" an online poll's results to fit their requirements, this is another example of an illegal use case. In this case, CAPTCHA security is used to prevent polling data entry. Regarding the legal ones, it can be a case where a new business partner wants to automate access to a service provided by a specific company but the service is CAPTCHA secured (to prevent abuse). However, the service provider has not yet made an Application Programming Interface (API) available for the new business partner to use. This may be because there is not enough time or money to make the API available. In this circumstance, the new business partner has no choice except to automate the CAPTCHA solving requirements. Time can be saved by avoiding pointless human tests using the CAPTCHA solution. Sites can be accessed without being barred. One does not have to pay for its service because it is free.

Overfitting is the primary potential drawback of utilizing a deep learning-based CAPTCHA solver. Overfitting is when a model performs well on training data but poorly on testing, validation, or unobserved data. In numerous examples, this has been seen in different applications.

### 1.4 Literature Survey

The study by Noury et. Al. featured text-based CAPTCHAs, which are popular but possess vulnerabilities that need to be identified. A deep neural network using customized convolutional layers and paralleled Softmax layer was used. Here, higher accuracy is achieved compared to CNNs with one Sigmoid layer. But the authors faced problems with variable length and alphanumeric CAPTCHAs [1]. Guerar et. Al studied the effectiveness of CAPTCHAs against constantly evolving malicious bots needs to be analysed to resist new bot attacks. Here, a summary of the present state-of-the-art in the field of CAPTCHA schemes has been provided, as well as a new classification system that encompasses all emerging schemes. Open issues, obstacles and potential for further research towards a more secure CAPTCHA generation have been highlighted. New CAPTCHA schemes and novel threats like human solver relay attacks, sensor manipulation and the risk of privacy breaches have not been considered [2].

Similarly, Tian et. Al. have looked into previous CAPTCHA breaking techniques require a high level of expertise and become useless as new security measures are introduced. A generic solver by combining unsupervised learning and representation learning to remove the noisy background of CAPTCHAs and solve text- based CAPTCHAs. Fewer labelled samples are required compared to deep learning methods. The method outperforms a fully- supervised model with the same network architecture in terms of recognition performance. The unsupervised decomposer is slower. CAPTCHAs with excessive character distortion and overlap are difficult to recognize. [3]

Another study features complex CAPTCHA schemes that can also possess vulnerabilities which need to be identified. A Depth First Search method to extract characters from CAPTCHAs by removing background noise and a CNN to recognize these extracted characters has been developed.

More resilient as security characteristics seen in CAPTCHA schemes like overlapping, rotation, distortion and waving have been integrated. It can be readily adjusted to variable length CAPTCHAs have not been solved. Here multi-word CAPTCHAs have not been solved. [4]

Ye et. Al. conducted a study using Machine learning based approaches need a vast amount of manually labelled real CAPTCHAs to train an effective CAPTCHA solver. A text CAPTCHA solver based on the generative adversarial network is built by first employing a CAPTCHA synthesizer to automatically generate synthetic CAPTCHAs to train a base solver and then using transfer learning to fine-tune the base solver on a small set of real CAPTCHAs. Much smaller set of real CAPTCHAs is required for training. Less human involvement is required when a new CAPTCHA scheme has to be targeted. It does not deal with variable length CAPTCHAs. But it does not solve multi- word CAPTCHAs. [5]

The study by Chen et. Al. is another interesting one in the same field. Existing Deep Convolutional Neural Network (DCNN) based CAPTCHA detection techniques have low identification performance in the confusion class. A two-stage DCNN that integrates all- class DCNN and confusion class DCNN along with a confusion class set partition algorithm have been proposed. It is a universal method that can be applied to other classifiers for object recognition. It has a higher success rate compared to existing CAPTCHA recognition methods. Fixed length CAPTCHAs have been used in training. [6] Wang et. Al similarly realized how traditional CAPTCHA recognition methods have low efficiency and poor accuracy. A new Dense Convolutional Network (DenseNet) for CAPTCHA recognition (DFCR) has been developed by decreasing the number of convolutional blocks and creating classifiers for various sorts of CAPTCHA images. Memory consumption is reduced. Better recognition performance compared to other DenseNet models. Datasets with fixed length CAPTCHAs were used for training. [7]

Azad et.al conducted a study in which the paper presents few of the most active attacks on text CAPTCHAs existing today.

The basic challenge in designing these obfuscating CAPTCHAs is to make them easy enough that users are not dissuaded from attempting a solution, yet still too difficult to solve using available computer vision algorithms. With advances in segmentation and Optical Character Recognition (OCR) technologies, the capability gap between humans and bots in recognizing distorted and connected characters becomes increasingly smaller. The emergence of recent advents and techniques made it more difficult to prevent automated bots and other dangerous spammers against CAPTCHA attacks. Some techniques we have discussed in this paper provide more than 40% success rate, and as the faulty CAPTCHA requests are reevaluated by the server and absence limiting count means that CAPTCHA decryption will be successful in consecutive attacks. [8]

A study by Burzstein et. Al. recognized how captchas are designed to be easy for humans but hard for machines. However, most recent research has focused only on making them hard for machines. They designed their study for two purposes: to collect information on the speed and accuracy with which humans solve captchas, and to collect information about a broad range of design and demographic factors that could potentially influence these results. They identified a number of demographic factors that have some influence on the difficulty of a captcha to a user. [9]

## 2. SYSTEM ARCHITECTURE AND PROPOSED WORK

### 2.1 System Architecture

We intend to use fully connected convolutional neural networks for the task of CAPTCHA detection. The initial layers of the model consist of 3 convolutional blocks, each consisting of a Conv2D layer with a different number of filters, BatchNormalization layer, MaxPooling2D layer and Activation layer. This is followed by a Flatten layer which is connected to our regressor dense neuron with SoftMax activation. As we intend to solve multiple lettered CAPTCHAs hence, we first propose training a regressor that can identify the number of letters in the image. Then we use the output of the regressor to detect the letters in the captcha by concatenating the regressor's output with the initial dense layer of our model. The final output layers consist of 6 dense layers, each with 37 neurons which corresponds to the number of unique characters. The following is the diagrammatic representation of our model:

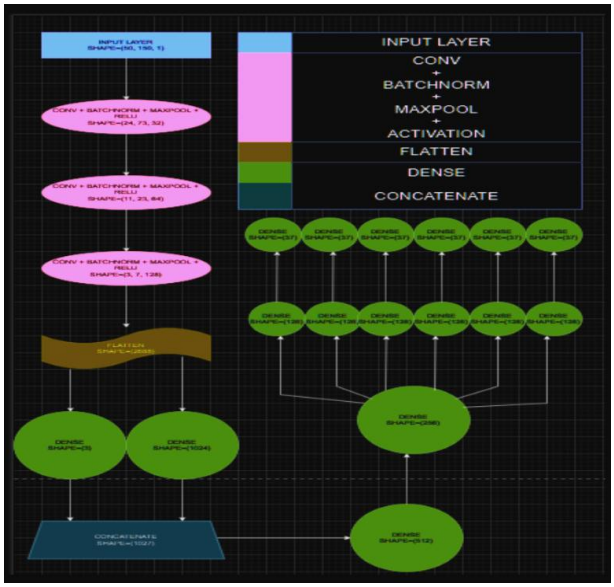


Figure 4. Architecture diagram of proposed model

The input layer takes in a NumPy array of shape (50, 150, 1). The next layer is a Conv2D layer with 32 filters and kernel size of (3, 5) and valid padding. This is followed by BatchNormalization layer, Maxpooling2D layer with pool size of 2 and finally relu activation layer. After these operations the dimensions change to (24, 73, 32). The following operations are repeated for a total of 3 times which gives the dimension of (3, 7, 128). These dimensions are then passed through a Flatten layer which converts the dimensions to (2688). Then we pass this to a Dense layer with 1 neuron which outputs the number of letters in the CAPTCHA image. The output from the regressor is Concatenated with the output of the Flatten layer which is passed through Dense layers with 512 and 256 neurons with some dropouts and activations. Then we use a for loop to output 6 streams (corresponding to the maximum number of letters in the dataset).

Each of these streams take input from the last dense layer with 256 neurons and pass it through a dense layer with 128 neurons. Finally, each stream has a dense layer with 37 neurons (corresponding to the number of unique letters) and SoftMax activation which gives a probability distribution of the letter it recognizes. Then, we compile the model using Adam as the optimizer and loss as categorical cross entropy as the outputs of the network are classes where each class corresponds to a letter at a position. Then we train the model for 100 epochs with callbacks like EarlyStopping, ModelCheckpoint and ReduceLROnPlateau.

### 2.2 Proposed Methodology: Algorithms Used

This consists of conducting surveys or getting information from various sources or using a pre- existing dataset or subset of a database. This is a crucial part of any machine learning application.

For Data pre-processing, this consists of understanding the type of data we are working with and performing various operations depending on the type of data. The various operations like normalization, resizing, reshaping, encoding, etc. For splitting into train, test and validation, this includes creating 3 different datasets either from one dataset or importing and pre-processing other datasets to create new datasets for testing our trained model and validating it during runtime. The optimal and most used train and validation split is 0.8-0.2 where 0.8 corresponds to 80% and 0.2 corresponds to 20% of total data respectively. For visualizing data distribution, this includes plotting or viewing our data using graphs, plots or metrics in order to gain an insight of the data. This also constitutes displaying images with corresponding task labels if working with image data. For constructing data pipeline, the purpose of data pipelining is to continuously feed our model with batches of data during training while keeping in mind to not overload the memory. Data pipelining is an important step when working with image data due to the size of each individual sample.

For constructing model architecture, this constitutes creating a map to direct the input through various layers and performing various operations like convolution, normalization, pooling, flatten, etc. The input and outputs should correspond to the inputs and outputs of the data pipeline. For creating custom loss and metrics, every deep learning model has losses and metrics. Most times they are prebuilt and do not need custom losses. But in some cases when the task to be achieved is different from traditional tasks, we need to define custom losses and metrics. They could be a mixture of pre-existing losses and metrics of completely new or user-defined. For compiling and fitting the model, this includes defining the losses and metrics and

passing data from the data pipeline to the model architecture. This is where the training takes place. For, plotting various metrics and losses obtained during training during training we get a bunch of values for the defined metrics and losses. These values change with every epoch. The losses usually reduce and metrics such as accuracy increase with epochs. There is a need to plot these values in order to understand our model performance. This gives us an insight to the various hyperparameters.

For testing the model on test data, this constitutes freezing the weights that are the learning block and passing the test data through the model and getting its performance in the form of losses and metrics defined. This also includes fine tuning some hyperparameters in order to gain better results. For, deployment, this stage occurs when we are satisfied with the training and testing results and want to put our model into production, that is the real world where it sees unseen data. This is done usually in the form of web apps, websites or mobile apps. Many platforms are available to deploy trained models.

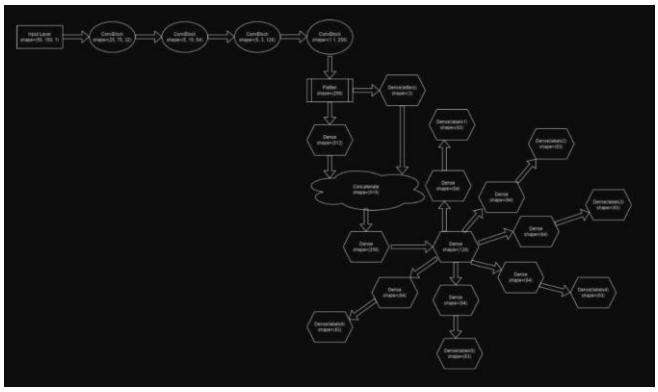

Figure 5. Diagrammatic representation of proposed methodology

### 2.3 Proposed Methodology: Performance Analysis

Our project is primarily based on the Deep Learning method. Its advantages and disadvantages include many. The advantages include how the features are automatically derived and tweaked for the best possible result, the many different applications and data types can, the benefit from the same neural network-based technique and how in the future, the deep learning architecture can be used to find the solution to new issues. The Disadvantages include how a massive amount of data is needed to outperform other method, how its complex data models are expensive to train and how topology, training technique and other parameters need to be well understood beforehand.

### 2.4 Results and Analysis

The model was trained for 100 epochs with callbacks like EarlyStopping, ReduceLROnPlateau and ModelCheckpoint. Due to this, the training stopped after 8 epochs due to no improvement in specified metrics. Each epoch took on average 715.25 seconds with average 15 millisecond per step. The learning rate reduced from 0.0001 to 0.00003 while training these 8 epochs due to the ReduceLROnPlateau callback.

The training loss at the end of the 8 epochs is 0.5543, training letters loss is 4.65x(10^-7), training labels0 loss is 0.1053, training labels1 loss is 0.0924, training labels2 loss is 0.1611, training labels3 loss is 0.1192, training labels4 loss is 0.0525, training labels5 loss is 0.0237, training letters accuracy is 1, labels0 accuracy is 0.9640, labels1 accuracy is 0.9690, labels2 accuracy is 0.9466, labels3 accuracy is 0.9590, labels4 accuracy is 0.9821, labels5 accuracy is 0.9921. The validation loss at the end of the 8 epochs is 26.8843, validation letters loss is 1.5252, validation labels0 loss is 6.2259, validation labels1 loss is 4.4652, validation labels2 loss is 5.2621, validation labels3 loss is 4.4762, validation labels4 loss is 2.0150, validation labels5 loss is 2.9146, validation letters accuracy is 0.7072, validation labels0 accuracy is 0.2597, validation labels1 accuracy is 0.3702, validation labels2 accuracy is 0.3481, validation labels3 accuracy is 0.2486, validation labels4 accuracy is 0.6409, validation labels5 accuracy is 0.7127.

The accuracy is varying because 5 different datasets have been used with three different number of letters. The letters loss is the error in the number of letters in the CAPTCHA image. The accuracy of the last two labels (label4 and label5) is higher as the number of 4-digit and 5-digit CAPTCHAs is greater than the total number of 6-digit CAPTCHAs.

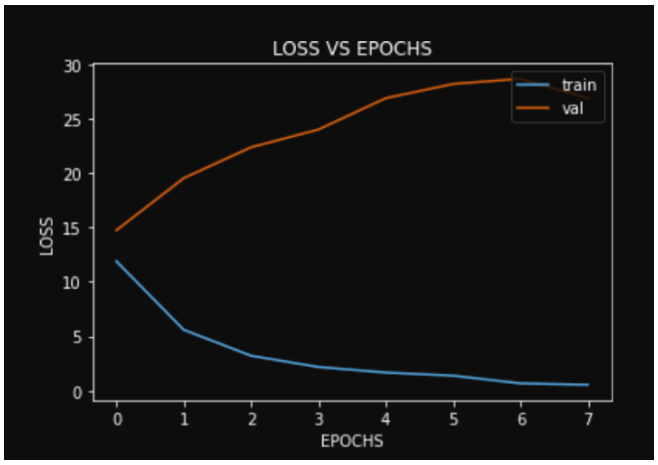


Figure 6. Total loss of the model vs. number of epochs

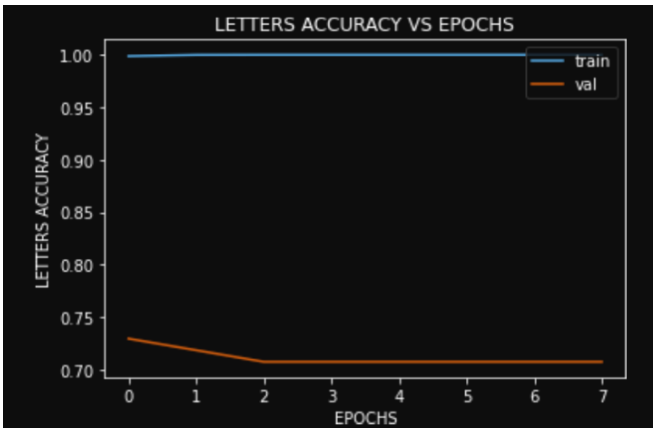


Figure 7. Total accuracy of the model vs. number of epochs

## 3. CONCLUSIONS AND FUTURE WORKS

We trained a neural network on various CAPTCHA image datasets that can guess the output of the CAPTCHA image we wish to break after image processing is applied to the image to extract the relevant information from it. We developed an effective CAPTCHA solver that can break text-based CAPTCHAs of any design without relying on hard-coded rules and constraints.

We pre-processed image datasets that was used to train our model. A data pipeline was created that helped in feeding image to our model. We used the pre-trained models like VGG16, MobileNet and DenseNet for the model and then trained our model accordingly. The results show that our model achieved high accuracy score for solving text-based CAPTCHAs, thus showing that our model can be used to find weaknesses in text-based CAPTCHAs so that they can be resolved in the future.

In the future, we intend to expand this project to identify characters of other language scripts too. We also wish to research effective techniques to break image-based CAPTCHAs too in the future. We will continue to train our model with bigger and more diverse datasets to improve the accuracy score further.

## REFERENCES

[1] Nouri, Z., & Rezaei, M. (2020). Deep-CAPTCHA: a deep learning based CAPTCHA solver for vulnerability assessment. Available at SSRN 3633354.

[2] Guerar, M., Verderame, L., Migliardi, M., Palmieri, F., & Merlo, A. (2021). Gotta CAPTCHA'Em all: a survey of 20 Years of the human-or-computer Dilemma. ACM Computing Surveys (CSUR), 54(9), 1-33.

[3] Tian, S., & Xiong, T. (2020, April). A generic solver combining unsupervised learning and representation learning for breaking text-based captchas. In Proceedings of The Web Conference 2020 (pp. 860-871).

[4] Atri, A., Bansal, A., Khari, M., & Vimal, S. (2022). De-CAPTCHA: A novel DFS based approach to solve CAPTCHA schemes. Computers & Electrical Engineering, 97, 107593.

[5] Ye, G., Tang, Z., Fang, D., Zhu, Z., Feng, Y., Xu, P., ... & Wang, Z. (2018, October). Yet another text captcha solver: A generative adversarial network based approach. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 332-348).

[6] Chen, J., Luo, X., Liu, Y., Wang, J., & Ma, Y. (2019). Selective learning confusion class for text- based CAPTCHA recognition. IEEE Access, 7, 22246-22259.

[7] Wang, J., Qin, J., Xiang, X., Tan, Y., & Pan, N. (2019). CAPTCHA recognition based on deep convolutional neural network. Math. Biosci. Eng, 16(5), 5851-5861.

[8] Azad S., & Jain, K. (2013). Captcha: Attacks and weaknesses against OCR technology. Global Journal of Computer Science and Technology

[9] Bursztein E., Bethard, S., Farby, C., Mitchell, J. C., & Jurafsky, D. (2010, May). How good are humans at solving CAPTCHAs? A large scale evaluation. In 2010 IEEE symposium on security and privacy (pp. 399-413). IEEE