MS Thesis Report

Robust CAPTCHA Solver using CNN (Convolutional Neural Network) based deep learning

A Thesis

By:

Zaki Masood
(NUST201463698MPNEC45314F)

Under Supervision of:

Dr. Naeem Abbas

Department of Electronics & Power Engineering
Pakistan Navy Engineering College (PNEC), NUST, Karachi

November 2017
MS Electrical (Communication) Engineering, Batch 2014

ABSTRACT

The CAPTCHA stands for Completely Automated Public Turing Tests to Tell Computers and Humans Apart, it distinguishes among human and machine by an image containing character or a picture, which computers directly cannot understand or decode without being programed [3]. CAPTCHAs provide answer which is easily responded by humans, but difficult for robots or machines. There are wide range of CAPTCHAs including text-based, visual-based and audio, but most commonly we used character based scheme [1, 2]. CNN is a framework that is used in many tasks like object classifications, automatic speech recognition or image processing. Proposed research uses an approach for solving character based CAPTCHAs based on CNN. However, a major requirement for training is a dataset, at start we collected data sample of 1571 text based CAPTCHA images, publically available on university of China's website. In this report we will concentrate on the detection of these CAPTCHAs using CNN. Initially we used simple CAPTCHA characters and achieved an accuracy of 96.3% and we achieved a success rate of 89.2% for the complex dataset. The results are verified using deep learning libraries of KERAS and TENSORFLOW, programmed in PYTHON language.

ACKNOWLEDGMENT

The accomplishment of the thesis from determination and devotion by various electrical faculty members of EPE department, and there provision stayed both direct and indirect throughout the task. I thank all of them for their devotion and generosity. Complementary thanks and acknowledgement to my thesis supervisor **Dr. Naeem Abbas**, for support and motivation throughout thesis work. And my GEC members **Dr. Syed Sajjad Haider Zaidi PN** and **Dr. Ali Hanzala Khan**, projected for being responsive and helpful throughout thesis work.

Above all, owe greatly tribute to the almighty Allah who provided us life and for benevolent us the power to complete work.

TABLE OF CONTENTS

Chapters Page No.

LIST OF FIGURES

LIST OF TABLES

Table No. Page No.

CHAPTER I

INTRODUCTION

1.1 An Overview to CAPTCHAs

Human uses their eyes and brain to visualize, and sense the planet around them. The field of computer vision, involves the understanding, analysis and automatic abstraction of useful data from a single image or a sequence of distinct images. It includes the progress of a hypothetical and algorithmic foundation to attain natural visuals. The term CAPTCHA was first invented by Langford et al in 2003 [1], CAPTCHA is indeed an acronym for Completely Automated Public Turing Test to distinguish human being and computers one from the other. What precisely is a Turing test? Alan Turing was a computer scholar who developed the Turing test which people use to check whether a machine can chitchat like a person [2]. A CAPTCHA is a transformed Turing test whereby a machine tests to check whether you are human or not. A computer or system anticipated to differentiate human being from the machine input, typically as a way of preventing spam and automated abstraction of data from the websites. Several categories of CAPTCHAs are being established including the text-based, image, audio, and video, however text-based CAPTCHAs as shown in Fig. 1.1 are most commonly used due to their easy applications. Presently researchers are using expertise of pattern recognition [3] in many applications such as speech recognition, face detection and data extraction of handwritten images [28]. Some of the techniques implemented using simple neural

networks, CNN based, supervised or unsupervised learning, HMM or SVM and so on others [4, 5].

Figure 1.1: Random CAPTCHA characters

CNNs are classification of neural networks that particularly powerful in face recognition and arrangement patterns [6]. CNN have been effective in object recognizing challenges and self-driving automobiles. CNN are comprised of layers with one or more convolutional layer(s) followed by fully connected layer. It contains neurons which possess weights and preferences and each neuron acquires little information sources. For CAPTCHA recognition we mainly divided goals into three major sections. First aim is to accumulate the dataset that includes testing and training images of characters. Initially we used 6284 characters from data sample of 1571 CAPTCHA images. Further we added noise and single horizontal line to make it complex, and used 10% of images for the testing and rest for the training. The next task is to establish a model which contains stack of CNN layers, we used two convolutional layers followed by pooling layer and then fully connected layer. Final stage uses weights and other relevant parameters that shows the character of higher percentage of confidence.

We shaped the model using two convolutional layer followed by pooling layer, trained on fixed length of CAPTCHA set of characters. In network after the layer of convolution, a max pooling layer is used, which applies non-overlying 2*2 matrix acquire maximum values of adjacent pixel. The input images fed into the network CAPTCHA character of two dimensional of size.

1.2 Motivation

In this report, method for solving particular character based CAPTCHAs, and is proposed using deep learning methodology which involves CNN (Convolutional Neural Network). Deep neural systems have numerous layers, where distinctive layers learn highlights at various levels. Bring down level layers learn low-level mechanisms, while higher layers learn larger amount highpoints. The whole system is prepared together in order to adjust the objectives of individual layers.

1.3 Why pattern based CAPTCHAs

Pattern based CAPTCHA tactics have some basic shortcomings that can be broken attackers. To start with, the name set, i.e., the quantity of classes, is settled. All naming based CAPTCHA plans require that the name set ought to be known to the client [13, 14]. To account human learning in addition making data more open, different schemes presently digitize somatic books which composed before the PC age. The pages of book are in effect photo examined, then afterward changed to content utilizing "OPTICAL CHARACTER RECOGNITION" or OCR. The change hooked on content is valuable, since inspection book gives images, that difficult to store proceeding tiny devices, costly for downloading, also not sought. The matter concerns OCR not much great. And

CAPTCHA improves way toward digitizing by referring words which cannot be examined by PCs. Mostly, separately letter which not scanned efficiently through OCR agreed to an image, then uses as CAPTCHA. Every fresh word which cannot perused effectively through OCR, forward to client in combination by additional letters which appropriate response is as of now known. Client formerly made a request to peruse mutually words. In event that they understand the solitary for which the appropriate response is known, the framework accept their answer is right for fresh individual. Framework at that time bounces new picture for various further individuals near decide, by advanced certainty, regardless of whether first response right. Table. 1.1 represents some character based CAPTCHAs which inspected through website, however agreeing to accept allowed email with Google, MSN/Hotmail, Yahoo! or Mail-blocks, in succession whose inquiry at Register.com or hunting down Ticketmaster (www.ticketmaster.com), and so forth.

1.4 DL (Deep Learning) Approach

CAPTCHAs that are conventionally characterized as logically developed issues, extremely hard to solve for AI calculations, however simple for the human being. Because of development in AI, an expanding number of CAPTCHA plans have turned out to be weak, as the basic AI issues ended up prominently reasonable by algorithmic machines [15]. In particular, late advances in Deep Learning (DL) decreased the space amongst human and machine capacity in taking care of issues that have been frequently utilized as a part of CAPTCHAs [1, 4]. To identify recognition of the CAPTCHAs the steps mainly divided into the three phases. First objective is to establish the model for

neural network which uses and have tendency to implement CNN (Convolutional Neural Network) layers. After the CNN compatibility the second objective is to accumulate and use the dataset for the recognition that includes the testing and training images which consists of numbers and letters. And lastly the final stage which consist of layers, weight and other relevant parameters and generates the result, to establish link, and program generated file which describes the module and contains results. Presently from the security point of view almost every website uses CAPTCHAs, which states that machine terminate accepting and cannot spontaneously read or decode with or without clearly programmed. The proposed research will recognize and give contribution that it is possible to read CAPTCHAs through machine learning using image processing and deep learning neural network.

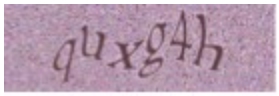
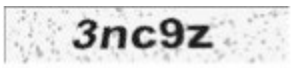
1.5 Character based CAPTCHAs

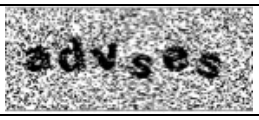
The study and research of the CAPTCHA identification will play vital towards improvement CAPTCHA security. Solving CAPTCHAs using deep learning will provide assistance in the identification to the difficulties related to present design, and heighten computer vision procedures associated by ICR. There are the following some commonly used CAPTCHA.

Table 1.1: Different CAPTCHA designs

Commonly Used CAPTCHAs





The proposed idea is described in this thesis report which involves seven chapters. Chapter. 2 includes why we need them and what they are, confers the background in the associated arena of CAPTCHA and how proposed research calculated. Chapter .3 covers the literature review section that involves basic understanding of CAPTCHA and DL methodologies such as CNN (Convolutional Neural Networks). Next is Chapter. 4 that discusses the structure implemented related to thesis work. Emphases on arranging and containing datasets in the Chapter. 5 for testing and training during research work. We tried distinguished models, and finalized our model and parts of it conferred as well. Simulated results for the training and validation are shown in Chapter. 6, and the last is Chapter. 7 which conclude the thesis report.

CHAPTER II

BACKGROUND

2.1 Related Work

Presently there is no unique segmentation algorithm that precisely segments all type of CAPTCHAs. Subject to the certain CAPTCHA characters comprising noises, however there have been established approaches towards and researchers develop methods. Hussain and Gao [7] also uses the subdivision of joined characters and recognize character using ANN (Artificial Neural Networks). Investigational outcomes have demonstrated 95.5% realization percentage. In 2014 Chen et al. [9], attained a typical recognition of CAPTCHAs comprising noisy dataset with the lines and they executed the probability design structure to identify CAPTCHAs, with success rate of 81.05%. They used quantitative approach to recognize CAPTCHA characters of target numbers, proposed a framework using probability pattern. Chandavale and Sapkal [10] presented a notable proceeding on the usability and security of CAPTCHAs, they exposed that colors essentially lessens the security as a replacement for refining it. And it likewise annoys the consumers sometimes, the usage of noise correspondingly marks the usability and security as an alternative of refining it. Presently from the security point of view website administrator uses CAPTCHA which states that machine terminate accepting and cannot spontaneously read or decode with or without clearly programmed. An early study by Morie et al. [11] mentioned that ez-gimpy which is a simplified version of the gimpy, CAPTCHAs are recognizable with the rate of 92%. They implemented the method, which identify the CAPTCHA character using context base matching for gimpy version.

Chellapilla and Simrad [12] also solved CAPTCHA by segmenting single characters and recognizing them, however in modern era of CAPTCHA design simple words cannot be segmented easily with simplified rectangular window because of overlapping of characters. Stark [8] contributed to solving certain dataset of text-based CAPTCHAs through deep learning, recognized that efficiency can be improved if we use small available training data, initially started with small dataset and by investigating their result to improve the classifier, and increase the training set including the test images, basis on their uncertainty.

In our model we used character segmentation and recognized CAPTCHA sets using deep convolutional network. Initially dataset was simple, we added salt and pepper noise and random lines to make it complex.

2.2 Neural Networks

The term neural network that establishes link between neuron and your brain, and particularly it is used to solve hard problems. It's a procedure for building a computer program that gains from some information or data. It is construct spontaneously in flowing of how we think the human mind functions. ANN stands for Artificial Neuron Network indeed a computational methodology grounded by arrangement in addition to tasks of biological network. Information which drifts from side to side the system interferes the configuration for the reason that system gets deviations - or absorbs, depending on the parameters of input and output. To understand neural network we need to understand the brain and it consist 100 billion cells and it is called neurons. In Fig. 2.1, these neuron are connected to pathways that actually transmit electrical pulses or signals, and cells gives neurons to have ability to send and receive electrical impulses [20]. The collective function is responsible for the function of brain, which in turn tells that brain is a mathematical function of some input and output. Each neuron has some input and output, when neuron see an electrical pulses from a cell it sends to other cell. A neural network is an attempt to computer to make model the brain [27].

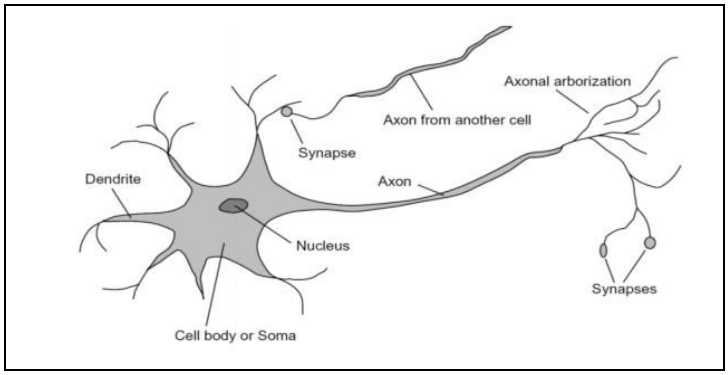


Figure 2.1: An example of biological neuron [27]

2.3 Supervised Learning vs Unsupervised Learning

Supervised or Controlled Learning kind of training that deals with related knowledge or information, which is functional the result that are right or not for solving computer vision problems [29]. Determine that uses system, modifies its boundaries as indicated by the training illustrations. As such, the preparation information ought to be named, which would help the system with taking in the parameters.

In unsupervised learning, it gains just from nearby data. It self-sorts out the information introduced to it and afterward identifies their properties. It separates the information into various groups speaking to comparable examples. This is especially helpful in spaces, where occasions are checked to coordinate past situations. For instance, in charge card extortion recognition, the example of a case can be coordinated with known misrepresentation designs.

On account of proposed CAPTCHA interpreter, we grouped the issue of CAPTCHA equally directed learning, including prepare cases ensure names allotted, so that our system figure out actual CAPTCHA character appears.

2.4 Convolutional Neural Network (CNN)

Convolutional Neural Networks are classification of neural networks that particularly powerful in some parts, for example, picture identification and arrangement. ConvNets have been effective in recognizing confronts, traffic signal and movement signs separated from fueling vision in robots and self-driving automobiles [31]. In Fig. 2.2, CNN (Convolutional Neural Networks) are build or comprised of layer with single or additional convolutional layer(s), either single or many fully connected layers. It contains neurons which possess weights and preferences [21]. Each neuron acquires little information sources, plays out fleck piece then instead backs it through nonlinearity.

The complete system silently communicates a lonely variable mark work: since simple image pixels near single horizontal near class notches at last. Despite everything it has adversity effort arranged layer former and every solitary of the slant made for learning general NN apply.

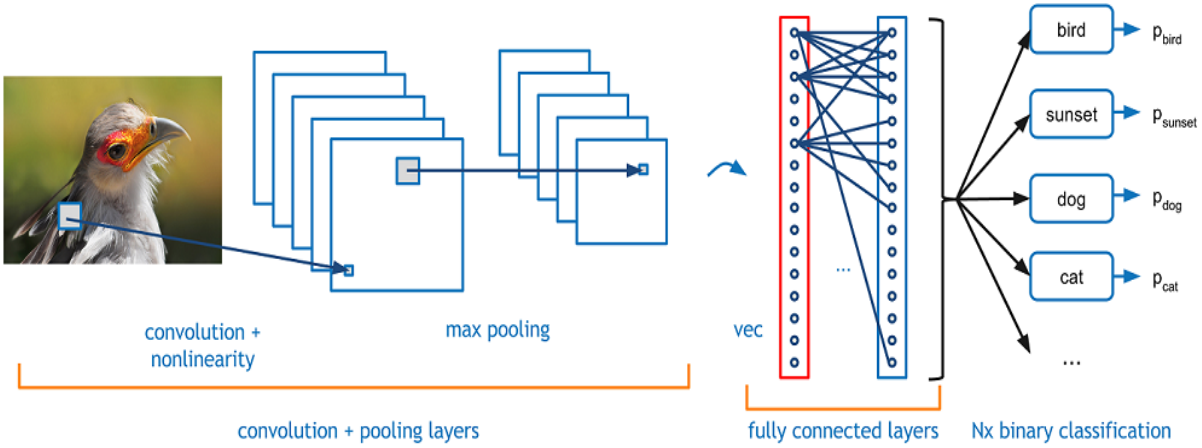


Figure 2.2: An example containing CNN Layers [34]

2.5 Architecture CNN

For the CNN approach that information involves images and sort design in a supplementary arranged method. Specifically, not at all like a normal NN, the layers of a CNN ensure neurons organized in three measurements containing height, width and last depth. Convolutional network consist several layers starting with convolutional and moving towards subsampling in its place later through absolutely connected layers. The straightforward architecture contains stack layers, each CNN layer changes single capacity of beginnings near other over a differentiable task. As shown in Fig. 2.3, Utilizing 3 major sorts of layers for construct CNN structures, first is Convolutional, then max-pooling and fully connected. The task is to connect such layers for frame a full structure of CNN's design. The impact for layers of convolution is m*m*r picture where m*m is image height and width respectively, and r contains number of channels, for example colors RGB of the image have r=3. A CNN structural design in modest case is a layer stack of that convert the picture bulk to an output volume.

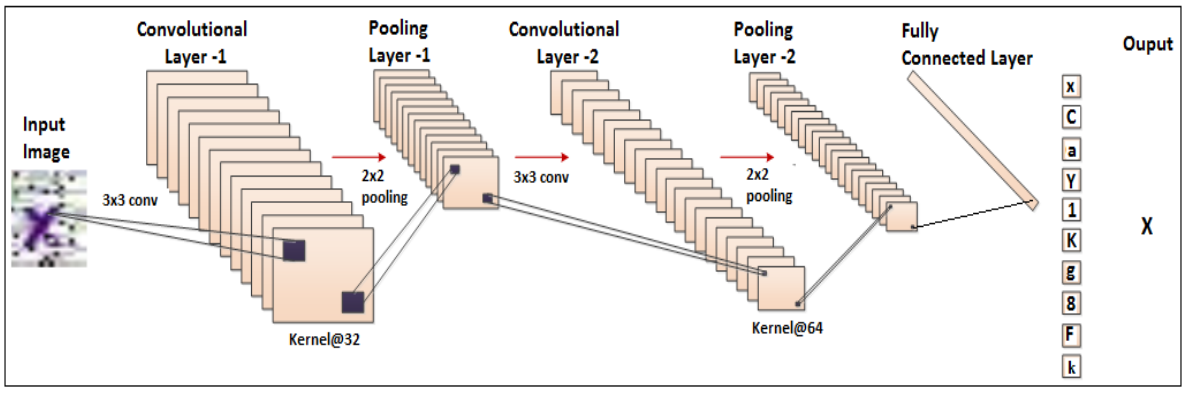


Figure 2.3: An Architecture block diagram of CNN, (designed in VISIO)

2.5.1 The Convolutional Layer

Convolutional layer applied to Fig. 2.4 yield of neurons which associated with nearby areas between input and information, individually figuring (.dot) product with item among their weights & minor local region associated within information. That might bring about volume, for example, [32x32x16] in the event that we chose to utilize 16 filters. Up till now we ensure solitary number. Keeping in mind, number i.e. quite recently illustrative while the channel is on upper left. If we replication same procedure aimed at each area of information (Subsequent stage desired to move channel one side through one unit, at that point right again with one, repetitively). Individually unique area of information of the input image creates a number. Subsequent to sliding the filter over every one of the location, you will discover that what you're left with for example 28 x 28 x 1 exhibit of numbers.

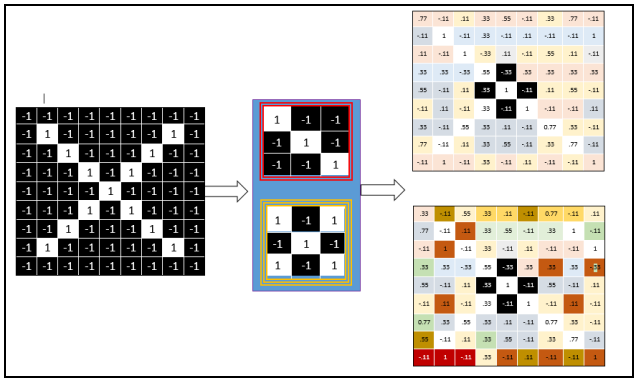


Figure 2.4: Applying filter for image feature using Convolutional Layer

They identify low level components, for example, edges and bends. As one would envision, with a specific end goal to anticipate whether a picture is a sort of question, we require the system to have the capacity to perceive larger amount elements, for example, ears or paws or hands. So how about we consider what the yield of the system is after the

primary convolutional layer. It would be a 28 x 28 x 3 volume. When experience another convolutional layer, the yield of the main convolutional layer turns into the contribution of the second convolutional layer. When we were discussing the primary layer, the information was recently the first picture. Be that as it may, when we're discussing the second convolutional layer, the information is the actuation map(s) that outcome from the primary layer. So each layer of the information is essentially portraying the areas in the first picture for where certain low level elements show up.

2.5.2 The Pooling Layer

One of the important slice of CNN include pooling layers, commonly connected subsequently convolutional. These subsample there information. And most well-known approach to do pooling it to apply a maximum operation to the consequence of each channel [22]. You don't really need to pool over the entire framework, you could likewise pool over a window. For instance, the accompanying shows max pooling for a 2×2 window in Fig. 2.5. Another critical behind CNNs associated to max-pooling that contains a type of indirect down-inspecting. Around a few non-direct capacities that execute pooling frequently max-pooling widely recognized. It segments info picture by an arrangement of non-top frames and aimed at individual sub-locale, yields greatest. The inclusion or pooling assists near logically lessen longitudinal portrayal scope, in the direction of decreasing quantity parameters, measure of system calculation which subsequently control over fitting. This one regular to occasionally embed pooling between progressive CNN layers into the design [18]. The pooling process gives certain type of interpretation consistency.

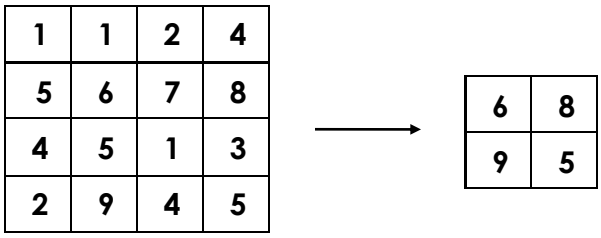


Figure 2.5: An example of max-filter with 2x2 using Pooling Layer

The pooling layer works autonomously on each profundity cut of the information and resizes. The most widely recognized is a pooling-layer containing filter or channel of size 2x2 connected with a moving step of 2 down examples at each profundity cut in the information by two laterally equally height and width. Adding further to the max pooling layer, pooling parts container correspondingly execute additional jobs, such as typical pooling. This pooling stood frequently used but has lately dropped out of approval likened to max-pool that ought to establish improved effort into training.

2.5.3 Fully Connected Layer

At long last, after a limited convolutional and pooling layers, irregular state present in system done through fully connected layers [25]. Neurons in connected layer need complete links by altogether enactments in previous layer, establish in consistent NN. There initiations subsequently figured by a grid duplication taken after by an inclination counterbalance. Final layer is called Fully Connected Layer, where every computed value from above layers gets a vote that results the true identification or what will be the answer have to be. The result are organized in single stream and in the form a list, on basis of these vote input image gets higher percentage will be identified. A template of all layers stacking together for character detection after final stage, is fully connected layer shown in Fig. 2.6, which identifies the right character at its output by “X” or “O”.

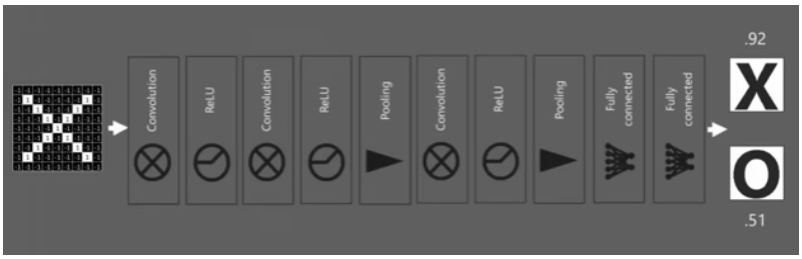


Figure 2.6: An Example of character detection after final stage fully connected layer [35]

CHAPTER III

LITERATURE REVIEW

3.1 Present CAPTCHA work

Since the beginning of the internet, clients have needed to make content incoherent to PCs. The primary such individuals could be programmers or hackers, presenting about slight points on online discussions they thought were by and large naturally checked for catchphrases [16]. To go around such channels, desired supplant single name with identical letters. For example "CALL" could move toward becoming "(/\|_|_" or "(/\££", and in addition various different variations, with the end goal that a channel couldn't in any way, shape or form identify every one of them. This later ended up noticeably known as leetspeak [5].

Currently text-based CAPTCHA's characters are charted to an extent that they require the concurrent utilization of three separate capacities—invariant acknowledgment, division or segmentation, and parsing to effectively whole the task with any consistency.

1. Invariant recognition refers to the ability to recognize the bulky variations in the shapes of letters. There are nearly an infinite number of versions for each character that a human brain can successfully identify. The same is not true for a computer, and teaching it to recognize all those differing formations is an extremely challenging task.

2. Segmentation, or the ability to separate one letter from another, is also made difficult in CAPTCHAs, as characters are crowded together with no white space in between.

3. Context is also critical. The CAPTCHA must be understood holistically to correctly identify each character. For example, in one segment of a CAPTCHA, a letter might look like an "m." Only when the whole word is taken into context does it become clear that it is a "u" and an "n."

Users have to tell exactly "what is it?" since label matching is strict. That is the reason why the set of alphabet letters and numbers is a popular choice. Note that the number of classes for alphabet letters and numbers are very limited. Due to deformation and noise in CAPTCHAs, the classes have to be further reduced to avoid confusion [17, 19]. Distortion often creates ambiguous characters, where users cannot be sure what they are. Although some characters have very different shapes, after distortion, they become hard to tell apart from each other. This problem is common in most character based schemes. We list common confusing character pairs as follows.

1. Letter vs digits: hard to tell distorted O from 0, 6 from G and b, 5 from S/s, 2 from Z/z, 1 from l.

2. Digit vs digit: 5 is hard to tell apart from 6, 7 is written differently in different countries and often what looks like a 7 may in fact be a 1, and 8 can look like 6 or 9.

3. Letter vs letters: Under some distortion, “vv” can resemble “w”; “cl” can resemble “d”; “nn” can could resemble “m”; “rn” can resemble “m” ; “rm” can resemble “nn”; “cm” can resemble “an”.

3.2 Activation function

Every one of the neurons in NN system include their information sources and place it in the capacity of non-linearity. Around wide range of sorts of enactment capacities utilized as a part of the scholarly community and engineering. The absolute maximum ordinarily utilized are sigmoid, stride work, ReLU, and hyperbolic.

3.3 ReLU Function

The ReLU stands for Rectified Linear Units. The primary reason that it is applied is a direct result of how effectively it can be figured contrasted with more regular representation like the sigmoid and hyperbolic deviation, without having a critical effect to accuracy. Normalization is to preserve mathematics since breaking through fine-tuning individual values just a bit, and change everything -ve to “0”. Layer for this normalization is done by using ReLU (Rectifier Layer Unit) layer.

The most generally carried output representation for conventional CNN neurons. Scientifically, it can be described:

$$F(x) = \max(0, x)$$

Sigmoid:

f(x)=1/ (1+exp(−x))

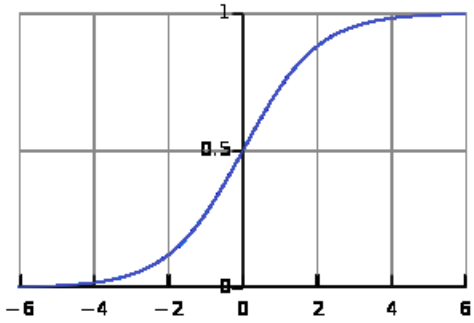


Figure 3.1: Sigmoid Function

3.4 Hyper Parameters

In Convolutional Neural Networks, the properties affecting arrangement layers and the neurons, as three-dimensional organization and concerned ground standards, termed as hyper-parameters. Hyper-parameters exclusively identify layers. The major parameters of CNN consider to be hyper-parameters are information dimensions (W*H*D) "W" is the Width, "H" and "D" is the Height and Depth respectively, and striding (S) and padding (P) and receptive field (R). In Table. 3.1, hyper-parameters for CNN are shown:

Table 3.1: Hyper-Parameters for CNN

Hyper-parameter	**Layer Name**
Input Dimension Volume	W*H*D
Striding-length	S
Zero-Padding	P
Receptive Field	R

3.4.1 Stride and Padding

Stroll controls how the filter convolves around the information volume. In the case we had to some extent 1, the channel convolves around the input bulk by moving one unit at once. The sum by which the filter movements is the walk. All things considered, the walk was verifiably set at 1. Walk is regularly set in a way so that the yield volume is a number and not a part. We should take a look at an illustration. For instance in Fig. 3.2, how about we envision a 7 x 7 input volume, a 3 x 3 channel and a walk of 1. This is the situation that we're usual to.

Along these lines, as should be obvious, the open field is moving by 2 units now and the yield volume contracts also. See that on the off chance that we attempted to set our walk to 3, then we'd have issues with dividing and ensuring the open fields fit on the information volume. Regularly, software engineers will expand the walk in the event that they need open fields to cover less and on the off chance that they need littler spatial measurements.

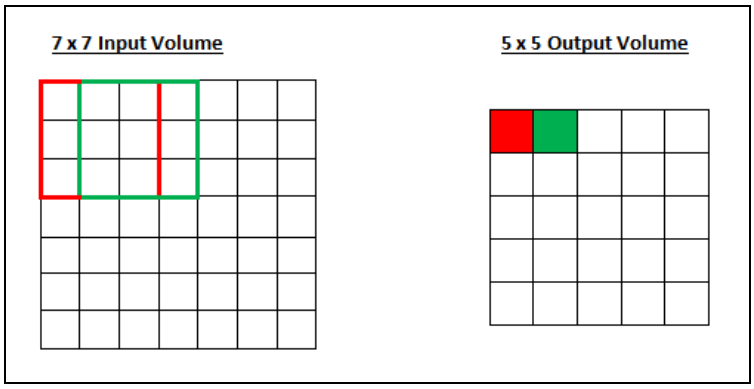


Figure 3.2: Striding and padding [34]

Presently, how about investigate mitigating. Before getting into that, we should consider a situation. What happens when you apply three 5 x 5 x 3 channels to a 32 x 32 x 3 input volume? The yield volume would be 28 x 28 x 3. See that the spatial measurements reduce. As we continue applying convolutional layers, the span of the volume will diminish speedier than we might want. Initial layers of system, need to save plentiful data around first info bulk with the goal that separate such little level elements. Suppose we need to apply the same convolutional layer however we need the yield volume to remain 32 x 32 x 3. To do this, we can apply a zero mitigating of size 2 to that layer. Zero suppressing the info volume with zeroes around the outskirt. On the off chance that we consider a zero suppressing of two, then this would bring about a 36 x 36 x 3 input volume [23].

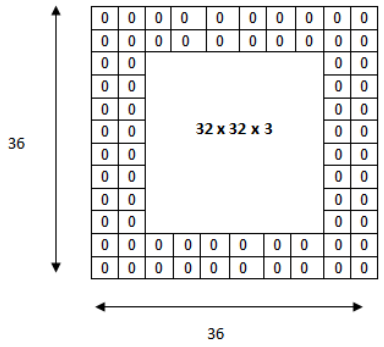


Figure 3.3: *Convolutional Layer and Filter

If you have a stride of 1 and if you set the size of zero padding to

$$Zero\ Padding = \frac{(K-1)}{2}$$

Where, K is the filter size, then I/O volume always ensure equivalent spatial dimensions.

The formula for calculating the output size for any given convolutional layer is

$$O = \frac{(W - K + 2P)}{S} + 1$$

Where O is the output height/length, W is the input height/length, K is the filter size, and striding (S) and padding (P).

3.5 Filtering

Filtering is to matching the feature of smaller pieces as shown in above Fig. 3.4, this filtration is done by using some mathematics behind and common steps for matching feature pieces uses:

1. Streaked feature and picture area.
2. Multiply individual picture pixel value by analogous feature pixel value.
3. Augment all.
4. Divide through all no. of pixel's value present in feature.

As can be analyzed in Fig. 3.4, the result generated by:

$$N = \frac{1 - 1 + 1 - 1 + 1 - 1 + 1 - 1 + 1}{9}$$

$$N = 0.111$$

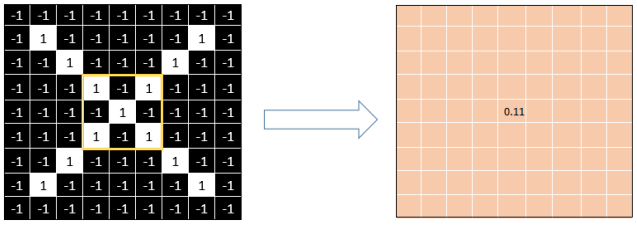


Figure 3.4: On left side filter is applied, right side contains its averaging value

For instance in above case each pixel value, and answer recorded in the corresponding position in Fig. 3.4. Moving the filter around the image we are getting actually the different values each time and recording these values, this is called mapping or feature matching. Further the moving, this filter over and over again onto every possible place is convolution This X is now convolved with different filter or feature of the image that resulting each time an output possibility. Thus one input image becomes stack of filtered images and is stored or arranged as shown in Fig. 3.5 containing output from two different filter on the image. This operation called Convolutional Layer operation, because it contains bunches of images and these images are stack, forming layer.

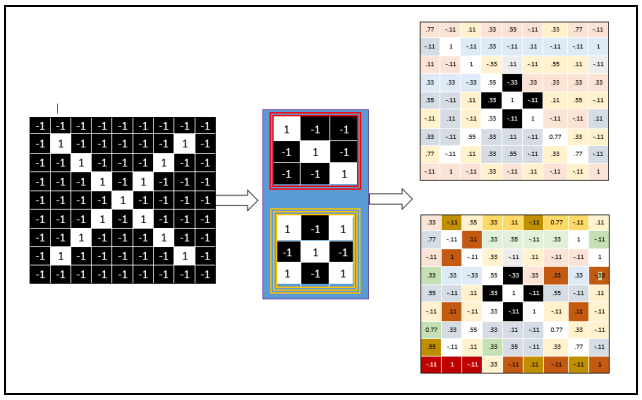


Figure 3.5: Output contains from two different filter on the image

3.6 Max-Pooling

The next step is to pooling, that is shrinking the image stack shown in Fig. 3.6 and common steps for pooling feature pieces uses:

1. Pick a window size (usually 2x2 pixel)
2. Choice striding length
3. Move channel through filtered pictures
4. Through individual window store maximum answer

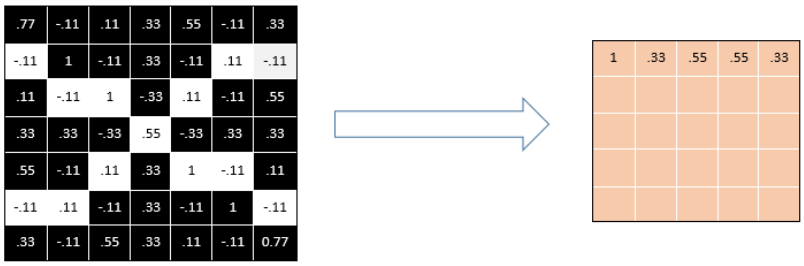


Figure 3.6: Windows maximum value

CHAPTER IV

DESIGN METHODOLOGY

4.1 An Overview

One successful approach for solving CAPTCHAs using deep learning is CNN, it is kind of structure, implemented in numerous supplementary task that includes object classification, spontaneous speech verification and computer vision applications [26, 30]. Though, main constraint aimed at training the network using CNN deep learning is bulky dataset [2]. Proposed research is related to deep learning and concentrate on the detection of particular category of CAPTCHA sets. It uses neural network system to process images that qualifies the learning network. For the proposed methodology to work competently, will be implemented with deep learning CNN (Convolutional Neural Network). CNN requires few prerequisites or dependencies that need to be considered, which includes model at the backend to support CNN deep learning neural network [32].

We used Convolutional Neural Network (CNN) to solve text based CAPTCHA. For the proposed research two convolutional layers are used, one fully connected and pooling layer are used. The system is made out using convolution and max pooling layers. The number of layers for convolution and max pooling finalized after achieving the optimized number of layers at the end of research. Two data set uses images, one for the training/learning and other testing the data for CAPTCHAs detection.

4.2 CAPTCHA Shape

Challenging for us in each time output should be accurate to identify true number or alphabet. What a computer is seeing and deciding on basis of 2-D array as can be seen in Fig. 4.1, our task now is to learn and train machine using deep learning. As shown in Fig. 4.2 computer compare the images pixel by pixel whether the values are equal or not if the values are equal it will identify as X otherwise not an X.

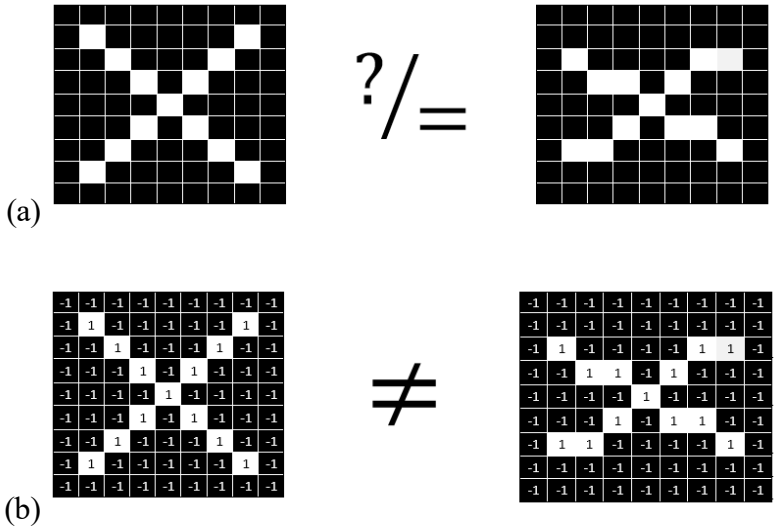


Figure 4.1: Left and right image to computer is not same

Now, what CNN (Convolutional Neural Network) does is to compare slice of an image parts by parts, or smaller number of features are compared. In Fig. 4.2 there are two windows are used on left and right side of the picture that are compared, rather to whole picture at once.

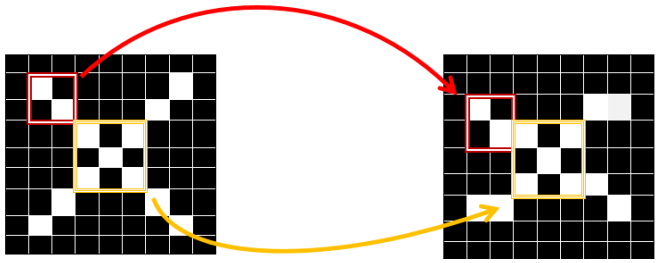


Figure 4.2: Comparing two different images

4.3 Character Recognition

We design and implemented a script to recognize individual characters, that includes some steps and these steps are shown in Fig. 4.3. And detail is summarize in the following steps:

a. Initially save CAPTCHA image and fed into the network on which we running our script or code.

b. The second step is to calculate the size of that CAPTCHA image. We find its window frame, 60 x 22 size in our case. To estimate the mean size of each individual character we divided image by four because we know that there are four character in CAPTCHA image. Which in turns give the individual character size as 15 x 22.

c. The individual character is now ready and sent to the classifier that quantifies the matching by already defined patterns of dataset character on which we trained network that will return confidence percentile.

d. The next task is to extract features from these sub window of individual characters and pattern values send to CNN classifier that matches these pattern to the predefined pattern generated during training associated feature values of one hot-vector of length 62 in array (.npy file).

e. At this is stage it is essential to classify individual character and store results. The leading percentile value for each character is taken out from classifier, and respective character's sub-window cropped from the original. And rest part of window saved and ready for another extraction.

f. Similarly the upcoming characters will be identified and process will continue till last CAPTCHA character.

g. The final step involves compiling results and show output containing four characters recognized on the basis of vote and retrieved with highest confidence are displayed.

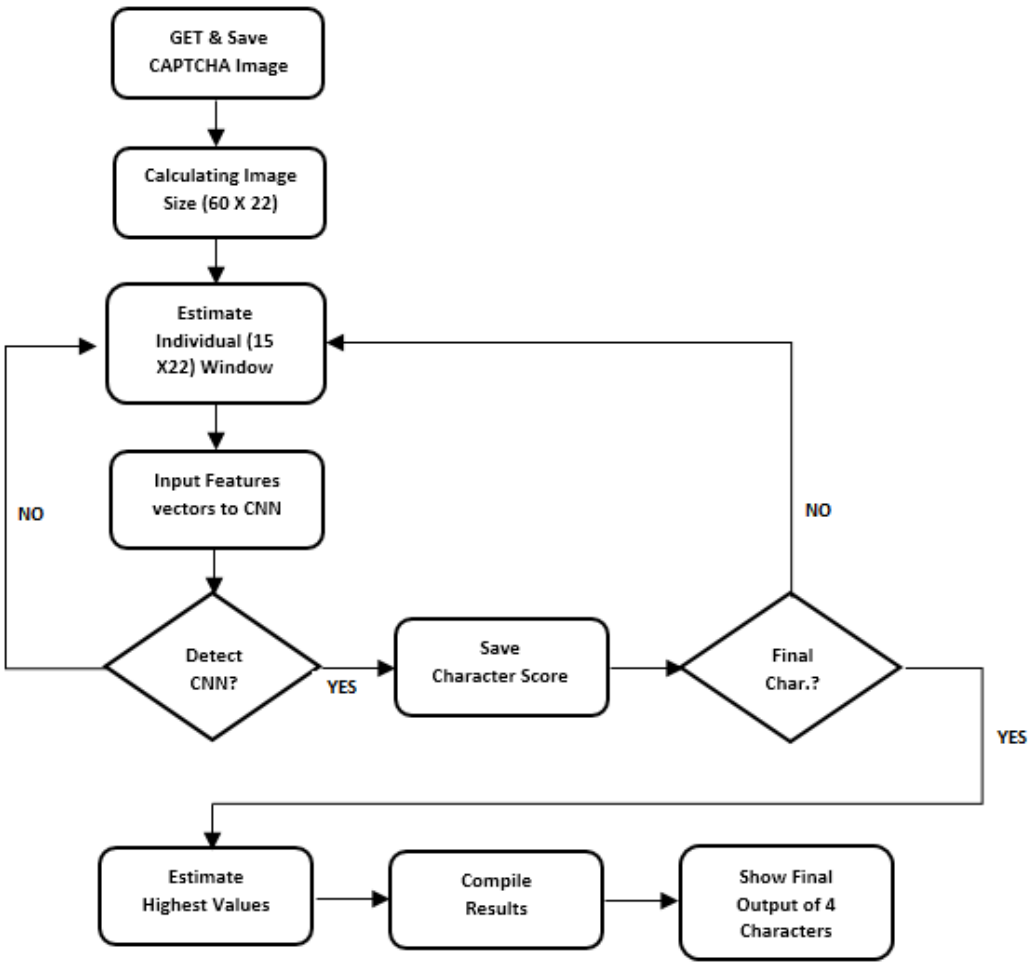


Figure: 4.3: Design Flow Chart

4.4 Recognition of Cropped Characters

There are total 62 characters which required to recognize and containing numbers, lowercase and uppercase letters based on CAPTCHA images. The first step is to normalize our dataset to a fixed size, as 15 x 22 in our case and each character is associated by featured value of one hot-vector of length 62 in array (.npy file). The CNN will train these 6284 data sample values and their associated vector lengths to train and 10% for testing the network. We tried different number of epochs during the training and testing period, after 500 epochs we simulated the final efficiency with 96.3% during training and 89.2% during testing the network.

4.5 CNN Layers Modeling

We shaped our model using convolution layer, ReLu, max-pooling, fully connected layer, and training on fixed length of pre-processed dataset of CAPTCHA characters. We also used dropout and softmax. For convolution layer, we operated with different number of convolution layers and finalized two layers. Selecting number of layers depending on input image size as mentioned earlier we will recognize 60 x 22 size of CAPTCHA, containing four characters with size of 15 x 22 and normalized. The variable for convolutional layers comprises convolution kernel and size of filter, in the first convolutional layer the kernel is 3x3 and there are 32 number of filters for feature extraction, which will generate output of 32 feature maps and followed by pooling with 2x2 max-pool and it will reduces the image size. In second convolutional layer we used same kernel of 3x3, but this time 64 number of filter for feature map. The ReLu f(x) = max(x, 0) is used instead of sigmoid as an activation and dropout layer with aspect of 0.25 that ultimately reduces complexity and help in minimizing time training process.

The final output is achieved with softmax layer to squashes output between 0 and 1 for all 62 characters.

a. Image Channel = 3 (RGB)
b. Convolutional layer-1: Size = 3x3, No. of filter = 32
c. Convolutional layer-2: Size = 3x3, No. of filter = 64
d. Pooling layer: Pool size = 2x2
e. Dense layer: No. of units: 512, in dense layer, ReLu as an activation function.

4.6 Design Model

Challenging for us that each time output should be correct to classify numbers or multi-font characters for all 62 classes. What a computer is seeing and deciding on basis of 2D array, our task is to train machine using deep learning by CNN. Computer compare the images pixel by pixel whether the values are equal or not if the values are equal it will identify as X otherwise not an X. Whereas CNN compare slice of an image parts by parts, or smaller number of features, there are also other steps involved. This X is now convolved with different filter or feature of the image, generating each time an output. Thus one input image becomes stack of filtered images and is stored or arranged. The operation is convolutional, because it contains bunch of images and these images are stacked, creating a layer. The results are organized in single stream of list, on basis of these vote, image characters that gets higher confidence percentage will be identified. We placed max-pooling layer to apply non-overlying 2*2 matrix to acquire maximum values of adjacent pixel. The input is a CAPTCHA character that fed into the network, width =

60 and height = 22, and single character after segmentation is of size 15 x 22 and are normalized to fed into CNN.

The difficult part is input image is not every time the same as in above case Fig. 4.1b, it changes and have different shape at its input as shown in next Fig. 4.4. The position can be shifted, sometime smaller, bigger or changed compared, which will then difficult to decide. It also may be thicker, thinner or rotated in shape in each case and its output possibility should be identified each time by CNN.

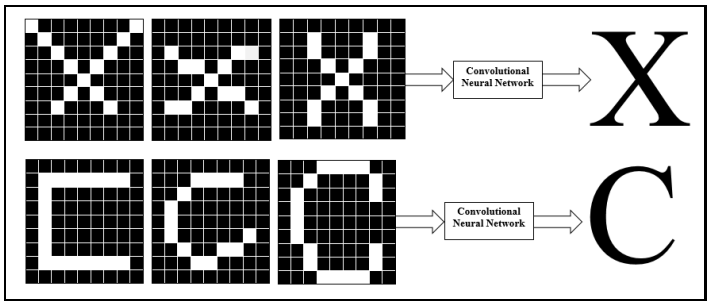


Figure 4.4: Different input image and CNN output

4.7 Stacking Layers

Now, stack of images now become small stack of images. Before final stage layer gets stack together and yields of 1st layer enters as input of 2nd, it continues for third. And resulting will be layer stacking, deep stacking may also be done by repeating these layering steps again and again. More will be the layers more will be the filter. Final layer is called Fully Connected Layer, where every computed value from above layers gets a vote that results the true identification or what will be the answer have to be [24]. The

result are organized in single stream and in the form a list, on basis of these vote input image gets higher percentage will be identified.

Table 4.1. Structure and Program

Structure	Program Platform
Language	PYTHON
Model	KERAS & TENSORFLOW
Operating System	Linux Ubuntu16.04
Operating Mode	CPU
Image Libraries	PIL, SCIPY, OPENCV

CHAPTER V

CAPTCHA DATASET

5.1 An Overview of Model

We shaped the model using two convolutional layer and two pooling layers, trained on fixed length of CAPTCHAs that is four digit. In network after each layer of convolution, a max pooling layer is used, which applies non-overlying 2*2 matrix towards acquire maximum values of adjacent pixel. The input images for the feed into the network are two dimensional of size height = 22 and width = 60.

5.2 Pre-Processing of Dataset

In this research, at the start we collected data sample of 1571 text based CAPTCHA images, publically available from the university of China. These data sample contains CAPTCHA characters having four digit with some low level of noises as shown in Fig. 5.1a. Further we added more salt and pepper noise and single horizontal line to make it complex Fig. 5.1b. The data is random and at present stage it is difficult to separate individual digits and categorize in list of numbers, small or capital characters. Since the dataset contains total 1571 images of sample CAPTCHAs, using manual cropping we cropped the images and random sample of these complex dataset are shown in Fig. 5.2. The dataset contains 62 characters including numbers starting from “0 to 9”, lowercase letters “a to z” and uppercase letters “A to Z”.

(a)

(b)

Figure 5.1: CAPTCHA complex image

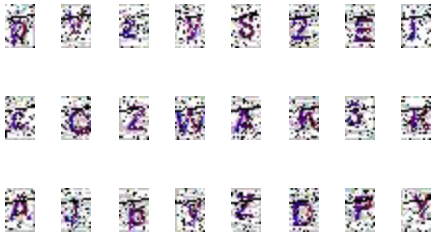


Figure 5.2: Segmented characters

5.3 Confusion Matrix

The Fig. 5.3 shows the confusion matrix for predictive analysis, since there are total number 6284 CAPTCHA character images, containing 934 for numeric characters and 3550 for multi-font characters. The multi-font characters further divided into 2687 small letter and 2663 for capital letters. We bundled 6284 images in single numpy array i.e. .npy file of all CAPTCHA character. The confusion matrix generated after testing CAPTCHA character images using our model and shows the results of actual input and predicted output. Total number of 242 characters out of 6284, which are recognized as wrong as shown in Table. 5.1, these number are distribution in non-diagonal rows of the confusion matrix. For instance in case of 934 numeric characters 34 wrong identified as small letters and 16 for capital letters.

	Predicted "0 to 9"	Predicted "a to z"	Predicted "A to Z"	Predicted %
Actual "0 to 9"	**884** 14.06%	**26** 2.78%	**22** 2.36%	**94.84%**
Actual "a to z"	**34** 1.27%	**2576** 40.99%	**59** 2.21%	**96.51%**
Actual "A to Z"	**16** 0.59%	**85** 3.16%	**2582** 41.08%	**96.23%**
Precision	**94.64%**	**95.86%**	**96.95%**	**96.14%**

Figure 5.3: CAPTCHA Confusion Matrix

Table 5.1: Confusion Matrix Table

Index Number	Alphabet or Number	No. of times wrong detection
17	h	7
29	t	4
30	u	6
28	s	20
15	f	7
40	E	3
58	W	4
5	5	21
14	e	3
6	6	9
8	8	7
54	S	10
12	c	5
33	x	8
43	I	1
26	q	5
55	T	8
27	r	6
60	Y	8
48	M	2
25	p	5
52	Q	1
3	3	2
20	k	2
57	V	9
16	g	4
34	y	3
38	C	4

31	v	7
42	G	5
51	P	10
7	7	3
41	F	2
59	X	5
9	9	7
2	2	1
35	z	8
19	j	3
11	b	2
10	a	4
46	K	1
32	w	3
13	d	2
61	Z	2
23	n	1
49	M	1
22	m	1
Total		**242**

CHAPTER VI

RESULTS

In result section we observe quantitative examination for the correctness of proposed design. The proposed network uses two convolutional layers and followed by max-pooling layers. Network exhibited KERAS and TENSORFLOW libraries, the results are achieved using equally simple and complex dataset of CAPTCHA images. There are total number of images used 6284 and 10% of the images used for testing of result. When we used simple dataset of 6284 images, succeeded accuracy rate 96.3% and 89.2% for complex dataset using individual CAPTCHA characters and 500 epochs as shown in Table. 6.2:

Table 6.1. Training Elements

Training Elements	Value
Number of epochs	500
Number of trained images	5027
Number of test images	1257
Accuracy simple CAPTCHA	96.3%
Accuracy complex CAPTCHA set	89%

The success rate of character recognition shown graphically using 500 epochs in Fig. 6.1. For testing we used 1257 of random character of images using altered dataset, during testing period, we started with lowest number of epochs, and by increasing we get high success rate. The Table. 6.2 depicts the accuracy achieved after testing is 89.2%. The confusion matrix summarizes the predicted rate of character classes which compares the numbers, lowercase and uppercase character of actual and predicted results. As mentioned in previous section we used dataset having high noise. If we use original dataset containing less noise then get higher success rate of recognition 96.3% with same number of epochs. These resulting outcomes are adequate to shield and substantiate that the text based CAPTCHAs are recognizable using machines.

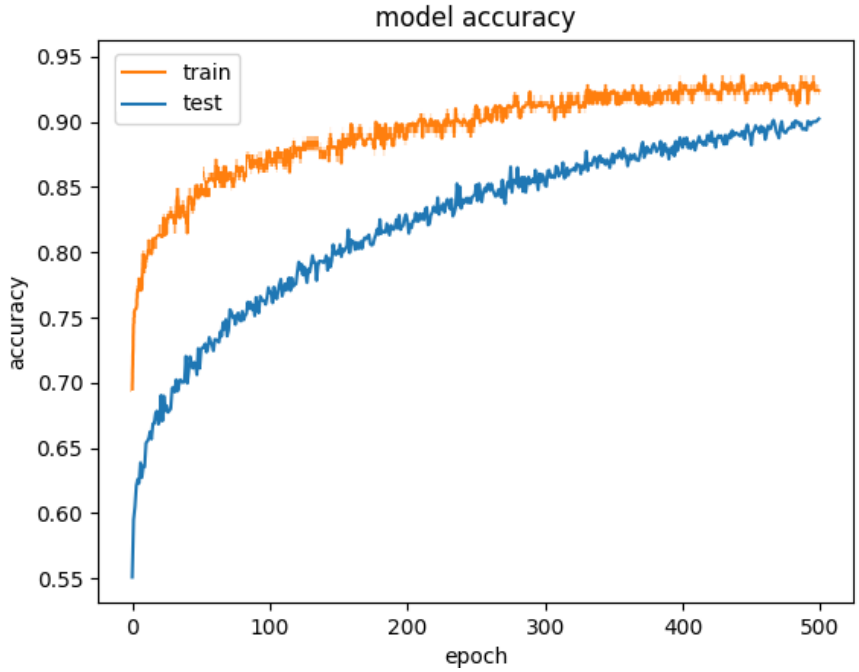


Figure 6.1: Training sequence accuracy vs. number of epochs

The success rate is achieved 89% using same number of 6284 images with complex dataset of CAPTCHA. Subsequently receiving a character correct is less demanding than getting the entire grouping right. The non-linear graph in Fig. 6.1 that indicates the accuracy of model, and accuracy is 89% during the training when using complex dataset is used. Training accuracy is plotted verses number of epochs, for training a model the more the number of epochs more it will take time, but accuracy can be improved. The exactness accomplished on settled length CAPTCHA was particularly notable, Table. 6.2 shows the comparing results with previous research related to character based CAPTCHA recognition using previous and latest approaches.

Table 6.2: Comparing Results with Existing Work

Existing Work	Accuracy	Year of Publication
Robust CAPTCHA Solver using CNN (Convolutional Neural Network) based Deep Learning	Success rate of 89% for complex dataset and 96% for simple dataset	2017
Ye. Wang. Proposed CAPTCHA is recognition, if and only if all of them have been correctly recognized.	The success rate of 85%[33]	2017
Hussain and Gao also uses the subdivision of joined characters and recognition by means of ANN	Investigational outcomes have demonstrated 95.5% realization percentage [7]	2016
In a research work by Chen et al., attained a typical recognition percentage	81.05% taking place CAPTCHAs comprising noisy dataset with the lines [9]	2014
Morie et al mentioned in a research paper that ez-gimpy which is a “simplified version of the gimpy”	CAPTCHA is also analyzable with the rate of 92% [11]	2003

These resulting outcomes are adequate to shield and substantiate that the text based CAPTCHAs are recognizable. Our structure further improves and suggest that CAPTCHAs that are text-based should be improved and are recognizable by bots or machines using deep learning models.

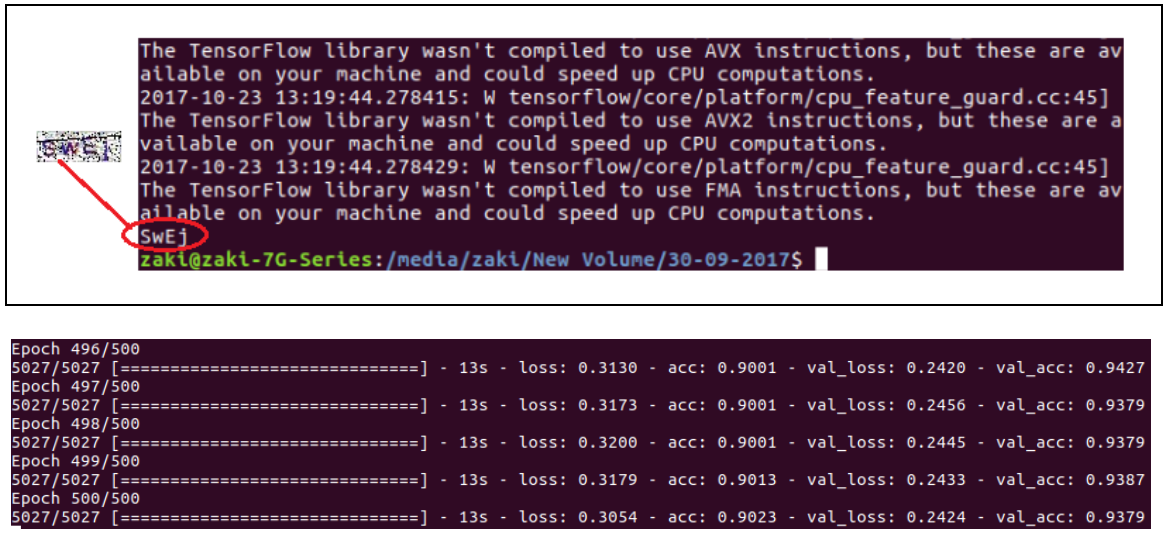


Figure 6.3: Accuracy and testing of Model

CHAPTER VII

CONCLUSION

In this report, we use convolutional neural network comprise of two convolutional layers followed by pooling layer, and shown results of character based CAPTCHAs that unravel an image constructed CAPTCHA utilizing deep neural network CNN. The result are generated with the dataset that is openly accessible with the goal that they can be utilized by other individuals for testing purpose. We have applied convolutional neural systems segmenting the picture, observing the individual characters. The model was prepared on both basic CAPTCHAs and complex CAPTCHAs after closing CAPTCHA characters together, further adding lines and noise in dataset. For the training concerns, we require range of information, so we used individual character containing simple and complex dataset of 6284 text based images.

The idea behind our research is to give comprehensive analysis on recognition and robustness of CAPTCHA characters. The investigation and research of CAPTCHA recognition using CNN will help to improve the glitches in present design and heightened growth of CAPTCHA security and related pattern. For future work, overlapped CAPTCHAs might be used.

REFERENCES

[1]. Ahn, Luis; Blum, Manuel; Hopper, Nicholas J.; Langford, John (May 2003). "CAPTCHA: Using Hard AI Problems for Security". EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques.

[2]. Andrew Cenkner; Vadim Bulitko; Marcia Spetch; Eric Legge; Craig G. Anderson; Matthew Brown "Passing a Hide-and-Seek Third-Person Turing Test" IEEE Transactions on Computational Intelligence and AI in Games Year: 2014, Volume: 6, Issue: 1 Pages: 18 – 30.

[3]. O.Starostenko, C. Cruz-Perez, F. Uceda-Ponga, & V. Alarcon- Aquino, "Breaking text-based CAPTCHAs with variable word and character orientation," Pattern Recognition, vol. 48, issue 4, pp. 1101-1112, Sep. 2014.

[4]. Yan, Jeff, and Ahmad Salah El Ahmad. "Breaking visual captchas with naive pattern recognition algorithms." Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual. IEEE, 2007.

[5]. Azizi Abdullah; Remco C. Veltkamp; Marco A. Wiering, 2009 " An Ensembl of Deep Support Vector Machines for Image categorization" International Conference of Soft Computing and Pattern Recognition, Year: 2009 pages: 301 – 306.

[6]. Alexis Vallet; Hiroyasu Sakamoto 2016, "Convolutional Recurrent Neural Networks for Better Image Understanding" International Conference on Digital Image Computing: Techniques and Applications (DICTA) Year: 2016 Pages: 1 – 7.

[7]. Rafaqat Hussain; Hui Gao; Riaz Ahmed Shaikh; Shazia Parveen Soomro. 2016,"Recognition based segmentation of connected characters in text based CAPTCHAs"8th IEEE International Conference on Communication Software and Networks (ICCSN) Year: 2016, Pages: 673 – 676.

[8]. Fabian Stark; Caner Hazrbus, Rudolph Triebel, and Denial Cremers 2015, "CAPTCHA Recognition with Active Deep Learning" Workshop New Challenges in Neural Competition Year: 2015 Pages: 99 – 102.

[9]. C. J. Chen, Y Wei, and W. P. Fang, "A Study on Captcha Recognition," In 10th international conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Kitakyushu, pp. 365-398, Aug.2014.

[10]. A.A. Chandavale, A. Sapkal, "Security Analysis of CAPTCHA," In Recent Trends in Computer Networks and Distributed Systems Security, vol. 335, pp. 97-109, Springer Berlin Heidelberg, 2012.

[11]. G. Mori, J. Malik, "recognizing objects in adversesarial clutter breaking a visual CAPTCHA," In proceddings of IEEE Computer Society Conference on Computer and Pattern Recognition in 2003, vol.1 pp.I 134–pp.I-141, 2003.

[12]. Chellapilla, K., Simrad, P.Y.: Using Machine Learning to break visual human interaction proofs (hips) ,In NIPS 2004.

[13]. Margarita Osadchy; Julio Hernandez-Castro; Stuart Gibson; Orr Dunkelman; Daniel P´erez Cabo 2016 "No Bot Expects the Deep CAPTCHA! Introducing Immutable Adversarial Examples with Applications to CAPTCHA" IACR Cryptology ePrint Archive Year-2016.

[14]. Suphannee Sivakorn; Iasonas Polakis; Angelos D. Keromytis 2016, "I am Robot: (Deep) Learning to Break Semantic Image CAPTCHAs" IEEE European Symposium on Security and Privacy (EuroS&P) Year: 2016 Pages: 388 – 403.

[15]. Chen Rui; Yang Jing; Hu Rong-gui; Huang Shu-guang 2013, "A Novel LSTM-RNN Decoding Algorithm in CAPTCHA Recognition" Third International Conference on Instrumentation, Measurement, Computer, Communication and Control Year: 2013 Pages: 766 – 771.

[16]. Oliver Watts; Gustav Eje Henter; Thomas Merritt; Zhizheng Wu; Simon King 2016 "From HMMS to DNNS: Where do the improvements come from?" IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).

[17]. R. Aadhirai; P J Sathish Kumar; S. Vishnupriya 2012, "Image CAPTCHA: Based on human understanding of real world distances" 4th International Conference on Intelligent Human Computer Interaction (IHCI) Year: 2012 Pages: 1 – 6.

[18]. L. Ahn, M. Blum and J. Langford. & quot; Telling Humans and Computers Apart Automatically. Communications of the ACM, 47(2):57–60, 2004.

[19]. G. Lv, "Recognition of Multi-Fontstyle Characters Based on Convolutional Neural Network," 2011 Fourth International Symposium on Computational Intelligence and Design, Hangzhou, 2011, pp. 223-225.

[20]. P. Yin et al., "A Novel Biologically-inspired Visual Cognition Model Automatic Extraction of Semantics, Formation of Integrated Concepts and Re-selection Features for Ambiguity," in IEEE Transactions on Cognitive and Developmental Systems vol. PP, no. 99, pp. 1-1.

[21]. Zejia Zheng, Zhu Li, A. Nagar and Kyungmo Park, "Compact deep neural networks for device based image classification," 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), Turin, 2015, pp. 1-6.

[22]. H. Dai Nguyen, A. D. Le and M. Nakagawa, "Deep neural networks for recognizing online handwritten mathematical symbols," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Kuala Lumpur, 2015, pp. 121-125.

[23]. S. Gutstein, O. Fuentes and E. Freudenthal, "Latent learning in deep neural nets," The 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, 2010, pp. 1-6.

[24]. D. C. Ciresan, U. Meier, L. M. Gambardella and J. Schmidhuber, "Convolutional Neural Network Committees for Handwritten Character Classification," 2011 International Conference on Document Analysis and Recognition, Beijing, 2011, pp. 1135-1139.

[25]. S. Dodge and L. Karam, "A Study and Comparison of Human and Deep Learning Recognition Performance under Visual Distortions," 2017 26th International Conference on Computer Communication and Networks (ICCCN), Vancouver, BC, Canada, 2017, pp. 1-7

[26]. X. Xu, J. Pan, Y. J. Zhang and M. H. Yang, "Motion Blur Kernel Estimation via Deep Learning," in IEEE Transactions on Image Processing, vol. PP, no. 99, pp. 1-1.

[27]. S. Kumar, A. Sharma, K. Mamun and T. Tsunoda, "A Deep Learning Approach for Motor Imagery EEG Signal Classification," 2016 3rd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), Nadi, 2016, pp. 34-39.

[28]. K. Shailaja and B. Anuradha, "Effective face recognition using deep learning based linear discriminant classification," 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Chennai, 2016, pp. 1-6.

[29]. F. Wu et al., "Weakly Semi-Supervised Deep Learning for Multi-Label Image Annotation," in IEEE Transactions on Big Data, vol. 1, no. 3, pp. 109-122, Sept. 1 2015.

[30]. R. Memisevic, "Deep learning: Architectures, algorithms, applications," 2015 IEEE Hot Chips 27 Symposium (HCS), Cupertino, CA, 2015, pp. 1-127.

[31]. Y. LeCun, "Deep learning & convolutional networks," 2015 IEEE Hot Chips 27 Symposium (HCS), Cupertino, CA, 2015, pp. 1-95.

[32]. Soniya, S. Paul and L. Singh, "A review on advances in deep learning," 2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI), Kanpur, 2015, pp. 1-6.

[33]. Y. Wang, Y. Huang, W. Zheng, Z. Zhou, D. Liu and M. Lu, "Combining convolutional neural network and self-adaptive algorithm to defeat synthetic multi-digit text-based CAPTCHA," 2017 IEEE International Conference on Industrial Technology (ICIT), Toronto, ON, 2017, pp. 980-985.

[34]. An example containing block diagram CNN (Convolutional Neural Network) Layers inhered https://adeshpande3.github.io/adeshpande3.github.io

[35]. An example of character detection after final stage of fully connected layer http://brohrer.github.io/how_convolutional_neural_networks_work.html