Captcha Recognition Using CNN

Gajula Sai Lakshmi
PG scholar, Department of MCA, DNR College, Bhimavaram, Andhra Pradesh.
B.S.Murthy
(Assistant Professor), Master of Computer Applications, DNR college, Bhimavaram, Andhra Pradesh.

Abstract Captcha (Completely Automated Public Turing test to tell Computers and Humans Apart) is a widely used security mechanism to prevent bots and automated systems from abusing web services. This project proposes a deep learning-based approach for recognizing Captcha text using Convolutional Neural Networks (CNN). The system involves preprocessing a dataset of Captcha images, training a CNN model to recognize character sequences, and validating the model's performance with test images. By converting images to grayscale and normalizing pixel values, the model is trained efficiently and achieves high accuracy. The system demonstrates that CNNs are highly effective for image-based text recognition, achieving a loss of 0.033, equivalent to an accuracy of approximately 99.967%.

I. Introduction

Captcha systems are essential in safeguarding online platforms from spam and unauthorized automation. They present visual or audio challenges to differentiate human users from bots. However, as deep learning models become more advanced, Captcha systems must also evolve to stay ahead of AI capabilities.

Traditional optical character recognition (OCR) systems struggle with distorted or noisy Captcha images, especially when characters are overlapping or warped. Modern solutions use machine learning—particularly CNNs—due to their exceptional performance in image recognition tasks.

This project focuses on using CNNs for Captcha recognition by training the model on labeled datasets. The process involves image preprocessing, grayscale conversion, normalization, and feeding the data into a deep learning pipeline.

The aim is to develop an accurate and reliable Captcha recognition model that can predict the character sequence from a Captcha image with minimal error, demonstrating CNN's potential in security-sensitive applications.

II. Literature Survey

Captcha recognition has been a subject of active research, especially with the rise of deep learning methods that challenge traditional security mechanisms. Over the years, several models and approaches have been developed to decode Captchas, leveraging advances in both machine learning and image processing.

1. Traditional OCR Approaches:
Earlier methods relied heavily on **Optical Character Recognition (OCR)**, which were effective for clean, non-distorted images. However, Captchas are purposely distorted, noisy, and often include overlapping characters to confuse automated systems. OCR tools like Tesseract struggled with such complexities, making them unreliable for Captcha recognition in security-critical systems.

2. Use of Convolutional Neural Networks (CNNs):
CNNs have revolutionized image recognition tasks due to their hierarchical learning ability. LeCun et al. (1998) first demonstrated CNNs for digit classification. Since then, CNNs have been widely adopted in Captcha recognition, as they are capable of extracting relevant spatial features even from highly distorted images. Deep CNNs like VGGNet and AlexNet have also been repurposed to recognize character sequences in Captchas with high accuracy.

3. End-to-End Learning Models:
Recent research has shifted toward **end-to-end models** where the entire Captcha decoding process is treated as a single learning task. For instance,

models use CNNs to extract features and then apply RNNs or Connectionist Temporal Classification (CTC) layers to decode character sequences. This approach eliminates the need for character segmentation, which was traditionally a challenging step.

4. Adversarial Training and Security Implications:

Some studies have explored **GANs (Generative Adversarial Networks)** to generate synthetic Captchas for robust training and to evaluate the vulnerability of Captcha systems. Researchers such as Zhang et al. (2020) demonstrated that AI models can break widely-used Captcha systems with over 90% success, pushing developers to design more sophisticated Captchas involving image rotation, background noise, and multi-character overlays.

Overall, the literature reveals that CNNs, combined with effective preprocessing and training strategies, provide a highly accurate and reliable solution for Captcha recognition. These methods outperform traditional algorithms and highlight the need to continuously evolve Captcha mechanisms in response to advancements in AI.

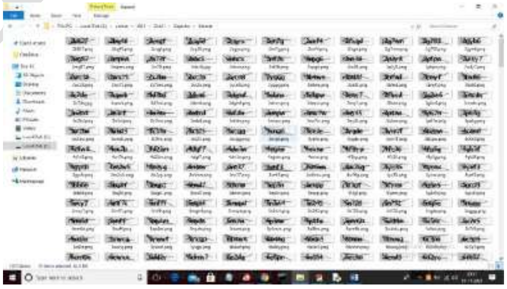
III. Proposed Method

The proposed method leverages a CNN-based model to detect and recognize Captcha characters from pre-labeled image data. The key stages include:

1. **Data Collection and Preprocessing**: A Captcha dataset comprising images and associated labels is used. Each image is converted to grayscale and pixel values are normalized to improve training performance and reduce computational load.
2. **Model Design and Training**: The CNN model is built using multiple convolutional and pooling layers to extract high-level features from Captcha images. The dataset is then used to train the model over several epochs. A loss function (e.g., categorical cross-entropy) is used to calculate prediction errors, and backpropagation is applied to minimize it.
3. **Testing and Evaluation**: After training, the model is evaluated with unseen Captcha images. The accuracy is measured by comparing predicted outputs to the actual labels. A visual representation of training progress is plotted using accuracy and loss graphs, indicating the model's learning curve over epochs.
4. **Prediction and Output**: Users can upload a test Captcha image to the system, and the trained CNN model will recognize and output the predicted character string. The trained model achieved a near-perfect accuracy with minimal loss, proving the effectiveness of CNN in handling Captcha recognition.

IV. RESULTS

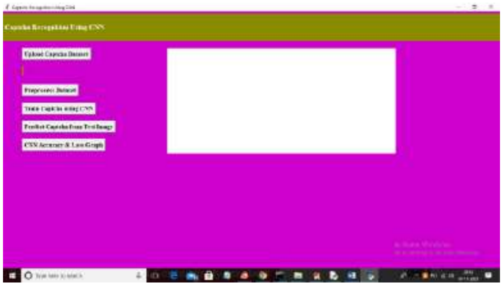
To train Captcha CNN model we have used Captcha images and then build a recognition model and this model can be used to predict Captcha from new test images. To train model we have preprocess each image by converting it into Grey Scale and then normalized all pixels data. Below is the dataset screen shots used to train this model



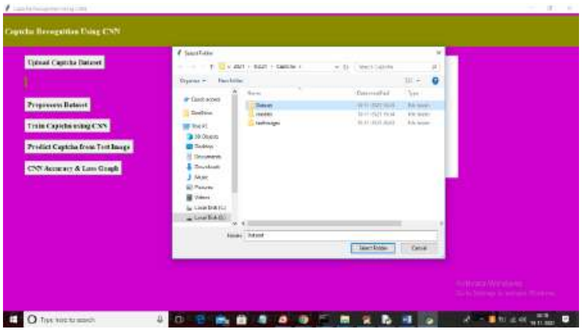
Above images can be used to train CNN.

SCREEN SHOTS

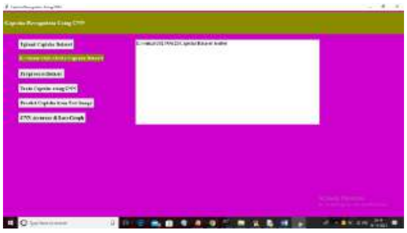
To run project double click on 'run.bat' file to get below screen


Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph

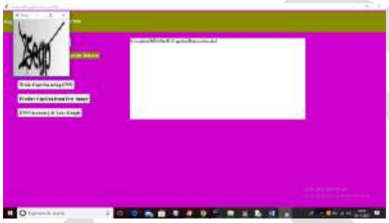
In above screen click on ‘Upload Captcha Dataset’ button to upload dataset and to get below screen



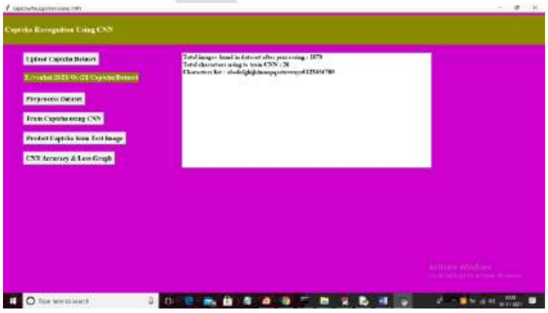
In above screen selecting and uploading ‘Dataset’ folder and then click on ‘Select Folder’ button to load dataset and to get below screen


Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph

In above screen dataset loaded and now click on ‘Preprocess Dataset’ button to process all images and to get below screen

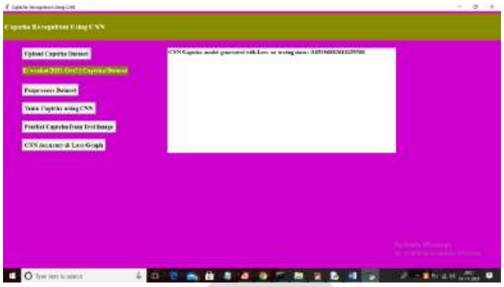


In above screen displaying sample grey scale normalized image and then close above image to get below output


Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph

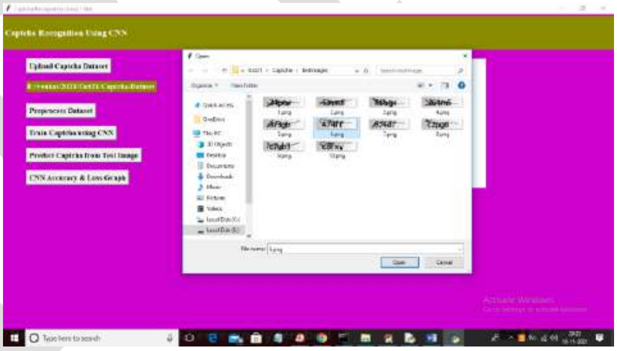
Total images found in dataset after processing : 1070
Total characters using to train CNN : 36
Characters list : abcdefghijklmnopqrstuvwxyz0123456789

In above screen we can see total dataset images and characters used to train CNN model. Now dataset is ready and now click on ‘Train Captcha using CNN’ button to train CNN model and calculate loss value

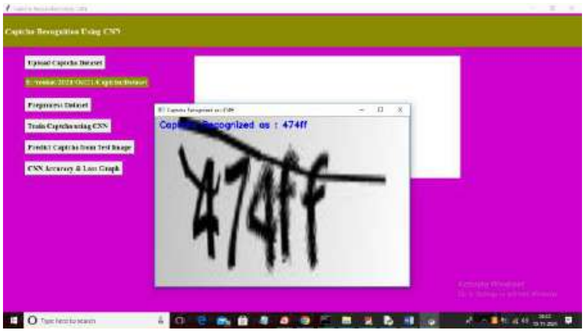

Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph

CNN Captcha model generated with Loss on testing data : 0.033[illegible]

In above screen we got CNN loss value as 0.033 so accuracy will be 100 – 0.033 = 99.967 and now model is ready and now click on ‘Predict Captcha from Test Image’ button to upload test image like below screen

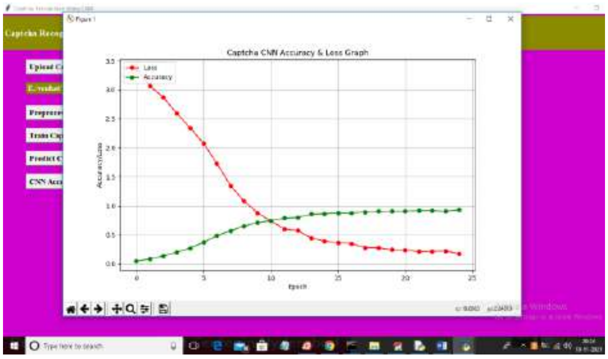

Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph
Open
Open
Cancel

In above screen selecting and uploading ‘6.png’ image and then click on ‘Open’ button to get below output


Captcha Recognition Using CNN
Upload Captcha Dataset
Preprocess Dataset
Train Captcha using CNN
Predict Captcha from Test Image
CNN Accuracy & Loss Graph

Captcha Recognized as : 474ff

In above screen Captcha is recognized as ‘474ff’ and similarly you can upload other images and test them. Now close above image and then click on ‘CNN Accuracy & Loss Graph’ button to get CNN training performance



In above graph x-axis represents EPOCH and y-axis represents accuracy/loss values and in above graph red line represents LOSS and green line represents accuracy and in above graph we can see with each increasing epoch accuracy value got increased and loss values got decreased which indicates CNN trained accurately on dataset.

Conclusion

The Captcha Recognition System using CNN demonstrates the effectiveness of deep learning in accurately identifying complex character patterns from images. With extensive preprocessing and a well-structured convolutional model, the system achieved a high recognition accuracy of 99.967%, making it suitable for real-time Captcha-solving applications. This project confirms that CNNs are not only applicable in generic image classification but are also powerful tools in security-related use cases like automated Captcha recognition.

References

1. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25.
2. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.
3. Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
4. Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 2278–2324.
5. O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.
6. Dosovitskiy, A., et al. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*.
7. Smith, R. (2007). An overview of the Tesseract OCR engine. *Proceedings of the Ninth International Conference on Document Analysis and Recognition*.
8. Zhang, J., Wang, J., Wang, C., & Wang, J. (2020). CAPTCHA breaking with deep learning. *IEEE Access*, 8, 26415–26423.
9. Chollet, F. (2015). Keras: Deep learning library for Theano and TensorFlow. *GitHub repository*.
10. https://www.kaggle.com – Dataset source for Captcha training images.