International Core Journal of Engineering Volume 11 Issue 7, 2025
ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

Captcha Recognition System based on Deep Learning

Ze Zhou, Quanzhu Yao*, Xinyu Liu

1School of Electronic Information, Xi Jing University, Xi'an, China

*Corresponding author

Abstract

In recent years, captcha recognition has emerged as a significant problem in the field of computer vision, aiming to enhance the machine's ability to recognize complex character images. It has widespread applications in areas such as network security and human-computer interaction. Traditional captcha recognition methods primarily rely on image processing and machine learning techniques. However, these methods exhibit certain limitations when dealing with complex captchas. Even a minor difference can lead to changes in their prediction results, thereby affecting the accuracy of captcha recognition and causing a significant drop in recognition accuracy. The advancement of deep learning technology has introduced new approaches and methods for captcha recognition. This paper proposes a deep learning-based captcha recognition system, which, by constructing complex validation sets and utilizing convolutional neural networks (convolutional neural network, CNN), trains on various types of captchas, effectively improving captcha recognition accuracy. Experimental results demonstrate that combining preprocessed datasets with CNN networks significantly enhances the stability of captcha recognition, achieving a recognition accuracy of over 95% and yielding promising experimental outcomes.

Keywords

Captcha Recognition; Convolution Neural Network; Data Preprocessing; Accuracy.

1. Introduction

CAPTCHAs are widely used in the field of the internet. Their original design intention was to distinguish human users from automated programs by generating complex images, text, or questions, thereby preventing machines from automatically logging into systems or performing other automated operations. They were first researched and invented by Luis von Ahn, Manuel Blum, and others at Carnegie Mellon University[1]. However, with the continuous advancement of deep learning and computer vision technologies, machines' ability to recognize CAPTCHAs has also been improving. This development not only poses challenges to existing CAPTCHA protection mechanisms but also provides new possibilities for the application of related technologies in other fields.

The goal of this research is to leverage deep learning techniques to enhance machines' ability to recognize complex character images. Although CAPTCHAs are designed to reduce the probability of machine recognition, improving machines' ability to recognize complex character images is of significant importance in certain application scenarios, such as handwritten font recognition and image text extraction. The advancement of such technology can not only improve the accuracy and efficiency of text recognition but also provide better information access methods for the blind and visually impaired. Therefore, this research will focus on exploring the application of deep learning in CAPTCHA recognition, aiming to improve machines' ability to recognize complex character images and investigate its potential value in other application scenarios.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

Early CAPTCHA recognition techniques primarily relied on traditional image processing methods, such as binarization, filtering, and edge detection. Although these methods are simple and easy to understand, they exhibit low recognition rates for complex CAPTCHAs and are highly susceptible to noise and interference. With the continuous progress of machine learning technologies, CAPTCHA recognition techniques have also begun to adopt machine learning-based methods. These methods mainly include support vector machines (SVM), decision trees, random forests, and others. In these approaches, machine learning algorithms can automatically recognize and classify characters or patterns in CAPTCHAs through the learning and training of large datasets, thereby improving the accuracy and efficiency of CAPTCHA recognition.

In 2003, Mori and Malik used shape context to recognize Gimpy and EZ-Gimpy CAPTCHAs, achieving recognition rates of 33% and 92%, respectively[2]. In 2008, Yan et al. successfully segmented Microsoft's CAPTCHA and achieved a recognition rate of 60% using multiple classifiers[3]. In 2005, Chellapilla et al. demonstrated that individual CAPTCHA characters could be well recognized by computers[4]. Scholars have also conducted research on CAPTCHA recognition based on segmentation methods. In 2007, Zhang Shuya et al. segmented and recognized SMT H-BBS CAPTCHAs, achieving recognition rates above 95% using K-nearest neighbors (KNN), BP neural networks, and SVM[5]. These recognition methods are tailored to specific types of CAPTCHAs and impose strict requirements on the standardization of CAPTCHAs, resulting in significant errors when recognizing different types of CAPTCHAs. With the ongoing development of computer technology, training neural network models for CAPTCHA recognition using deep learning-based approaches has improved recognition accuracy. Deep learning can adapt to different types of CAPTCHAs, eliminating the need for manual feature extraction. It also exhibits strong robustness to noise and interference through multi-level feature extraction and classification. By increasing the number of network layers and training data during the model construction process, recognition accuracy can be further improved, demonstrating strong scalability.

The experiment selected multiple combinations of CAPTCHA images and conducted repeated training using convolutional neural networks, achieving excellent experimental results. This approach effectively addresses the issues of limited CAPTCHA recognition types and low accuracy.

2. Model Architecture and Algorithm Design

2.1 Dataset Types

CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) is a technology used to verify user identity. It typically consists of a series of randomly generated characters or numbers that users must correctly input to pass the verification. The primary purpose of CAPTCHA is to prevent bots or malicious programs from automatically accessing or submitting data, thereby protecting the security of websites or applications.

This article focuses on image-based CAPTCHAs, where randomly generated characters or numbers are presented in the form of an image, and users are required to correctly identify and input these characters or numbers. For this project, a dataset of 15,000 images was used, divided into training, validation, and test sets. Specifically, the dataset includes:

- Training set: 7,500 images
- Validation set: 2,500 images
- Test set: 5,000 images

The dataset used for the CAPTCHA recognition system in this study is illustrated in Figure 1.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

Figure 1. Dataset for the Deep Learning-based CAPTCHA Recognition System.

2.2 Data Preprocessing

2.2.1. Image Binarization

Grayscale processing is a crucial step in image processing, primarily aimed at preparing the image for subsequent feature extraction operations, such as image recognition, segmentation, and medical image analysis. CAPTCHA images are typically in color, but deep learning models often require grayscale images as input. Therefore, the images are first converted to grayscale.

Following grayscale conversion, thresholding is applied to the grayscale image. In this process, the value of each pixel is compared to a defined threshold. Pixels with values above the threshold are converted to white, while those below the threshold are converted to black. Converting the image to a binary black-and-white format simplifies subsequent processing steps and enhances the model's ability to extract meaningful features. The binarization formula is as follows:

$$g(i,j) = \begin{cases} 0, f(i,j) > T \\ 1, f(i,j) \leqslant T \end{cases} \tag{1}$$

(T is a fixed threshold, obtained through the optimal threshold method.)

2.2.2. Noise Removal

CAPTCHA images often contain various types of noise, such as interfering lines, dots, speckles, and other artifacts. These noises can significantly degrade the performance of deep learning models by interfering with their ability to accurately recognize characters. Therefore, it is essential to preprocess the images to remove such noise.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

Several widely used noise removal techniques include median filtering, Gaussian filtering, and mean filtering. These methods help to enhance image clarity, eliminate noise, and improve the overall quality of the image, thereby facilitating better feature extraction and recognition by the model.

2.2.3. Character Segmentation

CAPTCHA images typically consist of multiple characters, which must be separated to enable individual recognition of each character. Character segmentation is a complex task that requires consideration of factors such as the distance between characters, their size, and their shape. To maximize the accuracy of the model during training, this study categorizes the characters into 31 classes, including:

- Digits: "2", "3", "4", "5", "6", "7", "8", "9"
- Letters: "A", "B", "C", "D", "E", "F", "G", "H", "J", "K", "L", "M", "N", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y"

To reduce ambiguity and improve recognition accuracy, 5 classes of characters that are easily confused by humans are excluded:"0", "1", "I", "O", "Z".

By selecting the remaining 31 classes as the output categories, the model achieves better training performance and higher recognition accuracy.

2.2.4. Character Normalization

After character segmentation, the size, shape, and position of each character may vary, which can negatively impact the recognition performance of the deep learning model. To ensure consistency in the size, shape, and position of each character, normalization is applied as a crucial preprocessing step. Common character normalization techniques include: Scaling, Rotation, Translation.

2.2.5. Data Augmentation

Data augmentation is a widely used method to effectively increase the volume of training data and enhance the generalization ability of deep learning models. There are various techniques for data augmentation, including rotation, translation, scaling, flipping, and adding noise. These methods generate additional training data, thereby improving the robustness of the model.

In this study, the preprocessed and segmented characters are categorized and visualized, as shown in Figure 2.

Figure 2. Display of CAPTCHA After Preprocessing, Segmentation, and Classification.

2.3 Model Design

Some deep learning models have been successfully applied to text, speech, and other recognition systems. In recent years, convolutional neural networks (CNNs), as one of the most important algorithms in deep learning, have demonstrated excellent feature extraction and generalization capabilities, achieving continuous breakthroughs in computer vision tasks such as image recognition and segmentation[6]. In this paper, convolutional neural networks are employed to effectively extract

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

features from CAPTCHA images through training. After the introduction of neural networks, researchers attempted to enhance the expressive power of models by increasing the number of network layers. However, this often leads to an excessive number of parameters in the model. When the scale of the dataset cannot match the number of parameters, the model tends to suffer from overfitting[7]. Inspired by the cognitive mechanisms of biological systems, researchers proposed convolutional neural networks (CNNs), which have shown remarkable performance when initially trained using the backpropagation (BP) algorithm. This is because CNNs reduce complexity by leveraging spatial relationships and weight sharing. In neural network models, the input layer refers to the feature vector of the input data, and the size of the input layer is consistent with the dimensionality of the feature vector[8].

2.3.1 Basic Structure of Convolutional Neural Networks

The training model in this study combines four convolutional layers, four pooling layers, and one fully connected layer to achieve efficient recognition and classification. The convolutional layers extract useful features from the images, and the accuracy of feature extraction directly impacts the recognition performance. The pooling layers then perform dimensionality reduction on these features, and finally, the fully connected layer matches these features to the output categories. By observing Figure 3, the basic composition of this convolutional neural network can be understood.

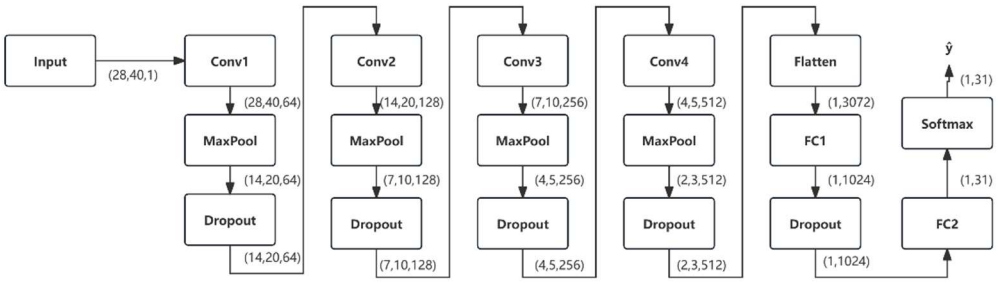

Figure 3. Convolutional Neural Network Design Model Diagram.

2.3.2 Convolutional Neural Network Design in CAPTCHA Recognition Systems

(1) Convolutional Layer

The convolutional layer consists of multiple convolutional kernels, each with its own weight parameters, designed to extract different features. The data parameters between convolutional units are adjusted using the backpropagation algorithm to achieve optimization. The first layer primarily extracts low-level features such as edges, while deeper layers extract more abstract and advanced features. The size and number of convolutional kernels significantly impact feature extraction. In this experiment, the parameters are set as follows:

- Conv1 channels: 64
- Conv2 channels: 128
- Conv3 channels: 256
- Conv4 channels: 512
- Convolutional kernel size: 3×3
- Input image size: 28×40
- Optimization method: Adam

(2) Activation Function

Activation functions are an essential component of CNNs, as they introduce non-linearity by mapping input signals to output signals. Commonly used activation functions in CNNs include ELU, LeakyReLU, ReLU, and Sigmoid, each playing a critical role in optimizing and enhancing neural network performance.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

Sigmoid: Characterized by a smooth S-shaped curve, it maps input signals to probabilities between 0 and 1. However, it suffers from the vanishing gradient problem.

ReLU: Its simple linear form effectively avoids the vanishing gradient issue but may cause neuron "death."

LeakyReLU and ELU: These are improvements over ReLU, further enhancing CNN performance.

In this study, ReLU and Softmax functions are used. The ReLU function truncates outputs less than 0 to 0 while keeping outputs greater than or equal to 0 unchanged. This sparsity reduces the number of parameters, mitigates the vanishing gradient problem, and accelerates the learning process. The Softmax function normalizes the output values in the final layer, ensuring that the probabilities of all classes sum to 1, thereby generating a probability distribution.

(3) Pooling Layer

Pooling layers[9] are commonly used in CNNs to reduce model size, improve computational efficiency, and enhance the robustness of feature extraction. Pooling layers reduce the dimensionality of feature maps, thereby lowering computational complexity. The size and stride of the pooling kernel affect the size and number of feature maps. In this study, max-pooling is employed with a 2×2 kernel and a stride of 2.

(4) Dropout Layer

The primary purpose of the Dropout layer[10] is to prevent overfitting in neural networks. Proposed by Hinton's team in 2012, this method improves the generalization ability of models[10]. Dropout randomly disconnects some neurons during training, reducing the number of active parameters. However, it has been observed that restoring all connections during inference yields the best results.

(5) Fully Connected Layer

The fully connected layer maps the extracted features to the corresponding output classes. The number of neurons in this layer significantly impacts model performance and can be adjusted based on specific requirements.

The framework of the CNN-based CAPTCHA recognition system includes: Preprocessing of input data, Design of convolutional layers, Design of pooling layers, Design of fully connected layers. In practical applications, adjustments and optimizations are necessary to improve the accuracy and efficiency of CAPTCHA recognition.

3. Experimental Testing and Analysis

3.1 Model Training and Testing

During the model training phase, the convolutional neural network (CNN) model is trained using the training set. A total of 10,000 CAPTCHA images are used for this training process. These images include both digits and letters, with a pixel size of 200×80. The dataset is divided into two parts:

- Training set: 7,500 images
- Validation set: 2,500 images

After completing the training, the model is tested using a separate test set of 5,000 images.

To improve the accuracy and robustness of the CAPTCHA recognition model, multiple hyperparameters are meticulously tuned and optimized during the training process.

3.1.1. Optimizer and Learning Rate Adjustment

Two commonly used optimizers, SGD (Stochastic Gradient Descent) and Adam, were tested under initial learning rates of 0.01, 0.001, and 0.0001.

When using the SGD optimizer with an initial learning rate of 0.01, the loss decreased rapidly in the early stages of training, but the final test set accuracy was only 84%. When the learning rate was lowered to 0.001 and 0.0001, the convergence speed further slowed, and the test set accuracy reached 94% and 92.3%, respectively. After switching to the Adam optimizer, which has the advantage of an

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

adaptive learning rate, Adam performed best with an initial learning rate of 0.001, achieving a test set accuracy of 98%. Further experimentation with a learning rate decay strategy, where the learning rate was halved every 10 epochs, showed that this strategy effectively prevented oscillations in the later stages of training, further stabilizing the model's performance, with Adam showing particularly strong results.

3.1.2. Depth and Width of Convolutional Layers

The initial model contains four convolutional layers, with 64, 128, 256, and 512 convolutional kernels, respectively. This structure performs well in capturing the complex features of CAPTCHA images, achieving a test set accuracy of 98%. However, when a fifth convolutional layer was added, using 1024 convolutional kernels, the model's representational capacity was increased. Despite this, overfitting occurred during training, leading to a slight decrease in test set accuracy (88%). At the same time, attempts to reduce the number of convolutional layers to three, with 64, 128, and 256 convolutional kernels per layer, resulted in faster training. However, due to insufficient representational capacity, the validation set accuracy dropped to 86%. Therefore, it was found that the four-layer convolutional structure performed best in capturing the complex characteristics of CAPTCHA.

3.1.3. Choice of Activation Function

In the four-layer convolutional structure, ReLU was used as the activation function, resulting in stable training and efficient computation, with a test set accuracy of 99.84%. This indicates that ReLU performs well in processing CAPTCHA images. After switching to the Leaky ReLU activation function, a small slope (α=0.01) was introduced to prevent the neuron "deadlock" problem. The validation set accuracy was 99.2%, which is close to ReLU, and slightly improved the stability during the early stages of training. Ultimately, ReLU was chosen as the activation function, as it demonstrated the best performance in terms of training stability and test set accuracy.

3.1.4. Dropout

To prevent overfitting, Dropout was introduced into the model. In the fully connected layers, Dropout with a retention probability of 0.9 was used, resulting in a test set accuracy of 99.84%. Lower retention probabilities led to excessive regularization, which negatively impacted the model's performance. Therefore, the retention probability was ultimately set to 0.9.

3.1.5. Batch Size and Training Time

Batch size has a significant impact on training speed and performance.

- Batch size 32: Test set accuracy is 93%, with longer training time.
- Batch size 64: Test set accuracy is 95%.
- Batch size 128: Test set accuracy is 99.84%, which is the optimal choice.
- Batch size 256: Test set accuracy decreases to 99.5%.

After continuously adjusting the model structure parameters, an accuracy of 99.84% was achieved. Figure 4 shows the loss function curve during the training and validation process of the final structure; Figure 5 shows the training accuracy curve during the training process. The learning rate used in training was 0.001, with a dropout value of 0.9. Training accuracy was computed and output every ten iterations, with a total of 1000 iterations. Figure 6 shows the accuracy of the model after testing. During accuracy calculation, the project displayed the index of the image from the entire dataset that was misclassified, with the misclassified CAPTCHA being **** and the correct CAPTCHA being ****. The accuracy calculation formula is = (Number of correctly classified images / Total images) * 100%.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

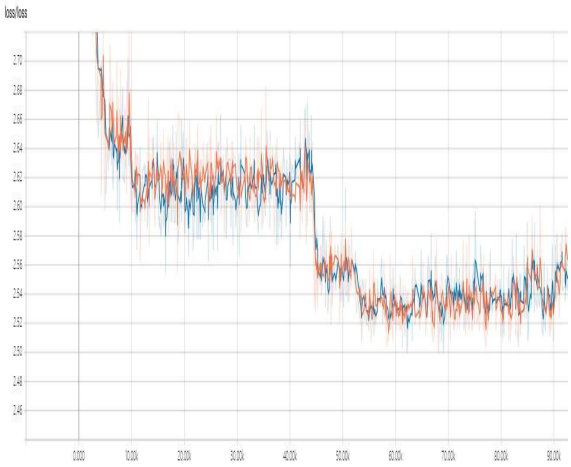

Figure 4. Loss Function During the Training and Validation Process of the Convolutional Neural Network Model.

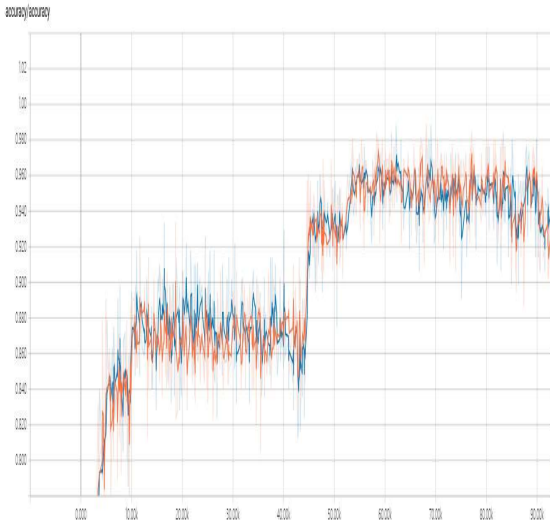

Figure 5. Accuracy Curve During the Training Process of the Convolutional Neural Network Model.

```
Number: 982 False: 6UYC True: 6UYU
Number: 3286 False: 7X36 True: 7N36
Number: 3335 False: R8WY True: R8WV
Number: 3785 False: FDMT True: FDMX
Number: 4443 False: MTXH True: MXXH
Number: 4471 False: 8BWC True: 8BWU
Accuracy: 0.9984
```

Figure 6. Test Accuracy After Training the Convolutional Neural Network Model.

3.2 Model Comparison

In the research, multiple experiments were conducted on the proposed deep learning-based CAPTCHA recognition model, and the ResNet-50 model was selected for comparison to comprehensively evaluate the model's performance.

Using the same dataset, the data was split into 70% training set and 30% test set. All algorithms were trained on the same training set and evaluated on the same test set. The evaluation metrics included

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

time cost (training time and inference time) and space cost (number of model parameters and memory usage).

In terms of model complexity, ResNet-50 is a classic deep residual network, consisting of a 50-layer deep network stack with a parameter count of 25.6M. In contrast, the CNN model significantly reduces the number of parameters by decreasing the depth of the convolutional layers and the number of neurons in the fully connected layers, resulting in only 8.1M parameters, about one-third of ResNet-50's parameter count. The main design strategy of the CNN model is to reduce the number of convolutional kernels in each layer while moderately increasing the use of pooling layers to reduce computation and thereby improve inference efficiency. The parameter comparison is shown in Table 1.

Table 1. ResNet-50 and CNN Model Parameters.

Model	Number of Layers	Number of Parameters(M)	Computational Complexity	Inference Time(ms)
CNN	13	8.1	1.8GFLOPs	18
ResNet-50	50	25.6	3.8GFOPs	28

As shown in the table, ResNet-50 has higher computational complexity and longer inference time, while the custom model, due to its fewer parameters and lower computational complexity, has a significant advantage in terms of inference time.

Both models were trained using the same image dataset and compared under the same hardware environment. With the optimizer settings and loss function being consistent, training time, accuracy on the validation set, and differences in convergence speed during training were recorded. The comparison results are shown in Table 2.

Table 2. Comparison of Results Between ResNet-50 and CNN Models.

Model	Number of Training Epochs	Validation Set Accuracy(%)	Convergence Speed (Epochs)	Training Time (Hours)
CNN	100	99.84	80	2.5
ResNet-50	100	99.87	120	4.0

From the results, it can be observed that the ResNet-50 model has a slightly higher accuracy on the validation set compared to the CNN model, but the CNN model converges faster and has a shorter training time. Due to its shallower structure and fewer parameters, the CNN model is able to find the optimal solution more quickly. Although ResNet-50 improves feature extraction capability in deep learning, its large network structure also leads to slower convergence and longer training times.

In addition to accuracy and training time, the memory requirements and computational efficiency of the two models during inference were also compared. The experiment measured GPU memory usage during the inference stage and the number of images processed per second (IPS). The efficiency comparison is shown in Table 3.

Table 3. Efficiency Comparison Between ResNet-50 and CNN Models.

Model	GPU Memory Usage(MB)	Inference Image Processing Speed (IPS)
CNN	850	230
ResNet-50	1550	170

ISSN: 2414-1895 DOI: 10.6919/ICJE.202507_11(7).0020

As shown in the table, the CNN model demonstrates a significant advantage in memory usage, requiring nearly half the GPU memory compared to ResNet-50. Additionally, in terms of inference speed, the CNN model is able to process more images per second due to its fewer parameters and lower computational complexity, thereby improving inference efficiency.

Through the above comparison and analysis, it is clear that the CNN model offers better performance in terms of training time, memory usage, and inference speed, making it suitable for applications that have higher hardware requirements or are sensitive to inference time. On the other hand, ResNet-50 has a slight advantage in feature extraction and accuracy, making it more suitable for larger-scale applications that require higher precision.

4. Conclusion

After discussing the background and significance of deep learning in CAPTCHA recognition systems, the research content was planned based on practical considerations, and an overall framework was established. By building a CNN model and training it on an existing dataset, and adjusting various hyperparameters, a comparison was made with the ResNet-50 algorithm. A deep analysis was conducted on its performance in terms of time cost, memory usage, and recognition accuracy. The experiments showed that the model's accuracy of up to 99.87% far surpassed other learning methods. At the same time, it achieved fast inference speed and low memory usage while maintaining high accuracy, demonstrating excellent performance. The results also confirmed the model's superiority in multiple key metrics, indicating good feasibility and effectiveness in practical applications. Future research will focus on further optimizing the model structure to improve recognition speed and accuracy, in order to meet the needs of a broader range of application scenarios.

References

[1] Ma Jianing. Research on Image CAPTCHA Recognition Algorithm Based on Deep Learning [D]. Shenyang Normal University, 2021.

[2] MORI G, MALIK J.Recognizing objects in adversarial clut-ter: breaking a visual CAPTCHA[C]//Computer Vision and Pattern Recognition.IEEE Computer Society Conference on.New York, USA: IEEE Press, 2003(1): 134-141.

[3] YAN J, A HMAD A S E.A Low-cost attack on a Microsoft CAPTCHA[C]//Proceedings of the15th ACM Conference on Computer and Communications Security.New York, USA: ACM Press, 2008: 543-554.

[4] CHELLAPILLA K, LARSON K, SIMARD P, et al.Comput-ers beat humans at single character recognition in reading based Human Interaction Proofs [C]//In Proceedings of the Second Conference on Email and Ant-i Spam.CA, USA: Stanford University, 2005.

[5] Zhang Shuya, Zhao Yiming, Zhao Xiaoyu, et al. Research on Character Recognition Methods for CAPTCHA [J]. Journal of Ningbo University: Natural Science and Engineering Edition, 2007, 12(4): 429-433.

[6] Li Wenjing, Bai Jing, Peng Bin, et al. A Survey of Graph Convolutional Neural Networks and Their Applications in Image Recognition [J]. Computer Engineering and Applications, 2023, 59(22): 15-35.

[7] Li Chuan. Research on Improved Neural Network Collaborative Filtering [D]. Xidian University, 2019.

[8] Xu Jiacheng. Design and Implementation of Convolutional Network Structure for Structured Feature Vector Input [D]. East China Normal University, 2019.

[9] Suarezpaniagua V,Segurabedmar I,Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction [J].2018.

[10] Yang Guanci, Yang Jing, Li Shaobo, et al. An Improved CNN Algorithm Based on Dropout and ADAM Optimizer [J]. Journal of Huazhong University of Science and Technology (Natural Science Edition), 2018, 46(7): 127-132.