



Research article

CAPTCHA recognition based on deep convolutional neural network

Jing Wang, Jiaohua Qin *, Xuyu Xiang, Yun Tan and Nan Pan

College of Computer Science and Information Technology, Central South University of Forestry and Technology, 498 shaoshan S Rd, Changsha, 410004, China

* **Correspondence:** Email: qinjiaohua@163.com.

Abstract: Aiming at the problems of low efficiency and poor accuracy of traditional CAPTCHA recognition methods, we have proposed a more efficient way based on deep convolutional neural network (CNN). The Dense Convolutional Network (DenseNet) has shown excellent classification performance which adopts cross-layer connection. Not only it effectively alleviates the vanishing-gradient problem, but also dramatically reduce the number of parameters. However, it also has caused great memory consumption. So we improve and construct a new DenseNet for CAPTCHA recognition (DFCR). Firstly, we reduce the number of convolutional blocks and build corresponding classifiers for different types of CAPTCHA images. Secondly, we input the CAPTCHA images of TFrecords format into the DFCR for model training. Finally, we test the Chinese or English CAPTCHAs experimentally with different numbers of characters. Experiments show that the new network not only keeps the primary performance advantages of the DenseNets but also effectively reduces the memory consumption. Furthermore, the recognition accuracy of CAPTCHA with the background noise and character adhesion is above 99.9%.

Keywords: CAPTCHA recognition; deep learning; convolutional neural network; DenseNet; ResNet

1. Introduction

To prevent the websites from being maliciously accessed by the automatic program in a short time and wasting network resources, the CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart) came into being. At present, the major websites have designed kinds of CAPTCHA with low resolution, multi-noise points, deformation characters, and adhesive characters. Therefore, designing a CAPTCHA recognition method can help to verify the security of existing various forms of CAPTCHA and assist in creating more robust CAPTCHAs. At the same time,

CAPTCHA recognition technology can also be applied in the area of license plate recognition, optical character recognition, handwriting recognition and so on. Scholars at home and abroad have done a lot of researches and got some progress in this field, which includes the traditional CAPTCHA recognition methods and the CAPTCHA recognition methods based on deep learning.

The traditional methods usually locate a single number or character area in an image and identify a single character after segmentation. For example, Lu Wang et al. [1] focused on the recognition of merged characters and proposed a method based on the local minimum and minimum projection values. It firstly divided the fuzzy-bonded characters and then combined the convolutional neural network (CNN) to identify a single character. But the recognition rate was only 38%. Yan et al. [2] successfully segmented the Microsoft CAPTCHA and recognized it by multiple classifiers, but the recognition rate was only 60%. Liang Zhang et al. [3] proposed a method based on LSTM (Long Short-Term Memory Network) recurrent neural network (RNN) for recognition. Long Yin et al. [4] suggested an approach based on dense scale-invariant feature transform (DENSE SIFT) and random sampling consistency algorithm (RANSAC), which had a recognition rate of 88% for simple sticky characters. It also had a good effect for the difficult twisted CAPTCHA. Hao Li et al. [5] proposed a Harris image matching method combining adaptive threshold and RANSAC. Lingyun Xiang et al. [6] used an adaptive binary arithmetic coding to encode English letters. After that, they also proposed a novel hashing method, called discrete multi-graph hashing [7]. Yuling Liu et al. [8] proposed a valid method for outsourced word segmentation, which saved storage space. Haitao Tang et al. [9] suggested a self-organizing incremental neural network based on PNN-SOINN-RBF. The overall prediction accuracy of single characters on the verification set of offline and online models were 72.75% and 50.25%. Yang Wang [10] et al. proposed a three-color denoising method based on RGB, using the method of segmentation character combined with contour difference projection and water droplet algorithm. This method had an excellent recognition effect on the CAPTCHA with background noise and character distortion adhesion. Yishan Chen et al. [11] proposed a method based on traditional digital image morphological processing technology for the segmentation and recognition of CAPTCHA, with the recognition rate of 60%. Ye Wang et al. [12] proposed a new adaptive algorithm to denoise and segment the CAPTCHA images, and used OCR (Optical Character Recognition) and template matching method to recognize a single character. Wentao Ma et al. [13] proposed an adaptive median filtering algorithm based on divide and conquer. Jinwei Wang et al. [14] proposed the CQWT-based forensics scheme for color images to distinguish CG and PG images. Inevitably, the above methods adopt manual data processing; there were three problems as follows:

(1) The way of direct segmentation for the adhesive CAPTCHA images is easy to cause character defects and aggravate the training task.

(2) Based on the global statistical feature or local feature descriptor of color, texture and shape, the extracted feature cannot accurately represent the images.

(3) Due to the imbalance of data, the results of classifier training are often not ideal and the selection of parameters adds a lot of difficulties for classifier training.

It was proved that a single feature could not adequately represent the image details [15]. The combination of both global and local features was used in image recognition method to achieve good performance [16]. With the rapid development and rise of artificial intelligence, the convolutional neural networks, by using the shared convolution kernel, have shown the effectiveness for multi-feature extraction and achieved excellent classification performance for two-dimensional graphs with invariant displacement, scaling and other forms of distortion. For example, Mingli Wen et al. [17] built

a CNN with only five layers, which achieved recognition accuracy as high as 99%. Peng Yu et al. [18] used the AlexNet for CAPTCHA recognition. After 20000 iterations, the recognition accuracy of this model was 99.43%. Zhang et al. [19] improved the LeNet-5, and the recognition accuracy was also as high as 95.2%. Shuren Zhou et al. [20] proposed a traffic sign recognition algorithm based on IVGG. Wei Fang et al. [21] proposed an image recognition model based on CNN, which can enhance the classification model and effectively improve the accuracy of image recognition. Lv Yanping et al. [22] used CNN to identify Chinese CAPTCHA images with distortion, rotation, and background noise. Garg and Pollett [23] developed a single neural network capable of breaking all character-based CAPTCHA. Yunhang Shen et al. [24] proposed a structural model based on multi-scale Angle to identify the currently popular Touclick Chinese CAPTCHA images. Wang Fan et al. [25] used the Keras framework for building CNN to identify Chinese CAPTCHA images with an accuracy of 92.8%. Directly input the CAPTCHA images into the trained CNN to identify which can effectively simplify manual intervention such as character segmentation, position and noise problems. Overall our main contributions are as follows: (1) we propose a CAPTCHA recognition method based on the deep CNN. By identifying different types of CAPTCHA images, it can improve the recognition accuracy and provide a convenient way for the website users to verify the security of their CAPTCHAs; (2) we design a new DenseNet that effectively reduces the memory consumption and show excellent performance.

2. DenseNet model

In 2017, Gao Huang and Zhuang Liu et al. [26] constructed 4 deep CNN called DenseNets, which connected every two layers in the network with “skip-connection.” It means that the input of each layer is the union of the output of all the preceding layers, which is different from the traditional network where each layer is only connected to the subsequent layers. The DenseNets have several compelling advantages: they solve the problem of gradient dispersion, and effectively utilize the features of all the preceding convolutional layers, which reduce the computational complexity of the network parameters and show excellent classification performance.

$$x_l = H_l([x_1, x_2, \dots, x_{l-1}]) \quad (1)$$

In Eq 1, x_l indicates that the l^{th} layer received the feature mapping of all the preceding convolutional layers as input, and $[x_1, x_2, \dots, x_{l-1}]$ is a tensor referred to the concatenation of the feature-maps. Therefore, even the last layer can receive the output of the first layer as input. As shown in Figure 1, Gao Huang inputs a given image into the DenseNets, then the network will predict the classification result after convolution and pooling in three Dense Blocks.

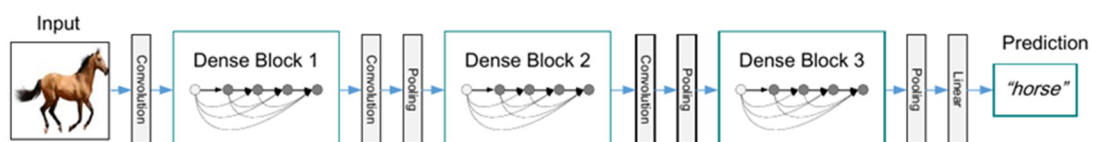


Figure 1. A deep DenseNet with three dense blocks.

Gao Huang and Zhuang Liu have designed 4 networks, such as DenseNet-121, DenseNet-169, DenseNet-201, and DenseNet-264. However, all the DenseNets' convolutional groups are 12 times in Dense Block 2. But Ma N et al. [27] proved that convolutional groups increased the complexity of the network and occupied much memory resources. Therefore, we improve the structural of DenseNets and propose a CAPTCHA identification method based on the DenseNets.

3. DenseNet model construction for CAPTCHA recognition

Based on the architecture of the DenseNets, we build a new deep CNN called the DFCR.

Firstly, the original CAPTCHA images with a size of 224×224 are convoluted and pooled to output the cropped CAPTCHA images with a size of 56×56 .

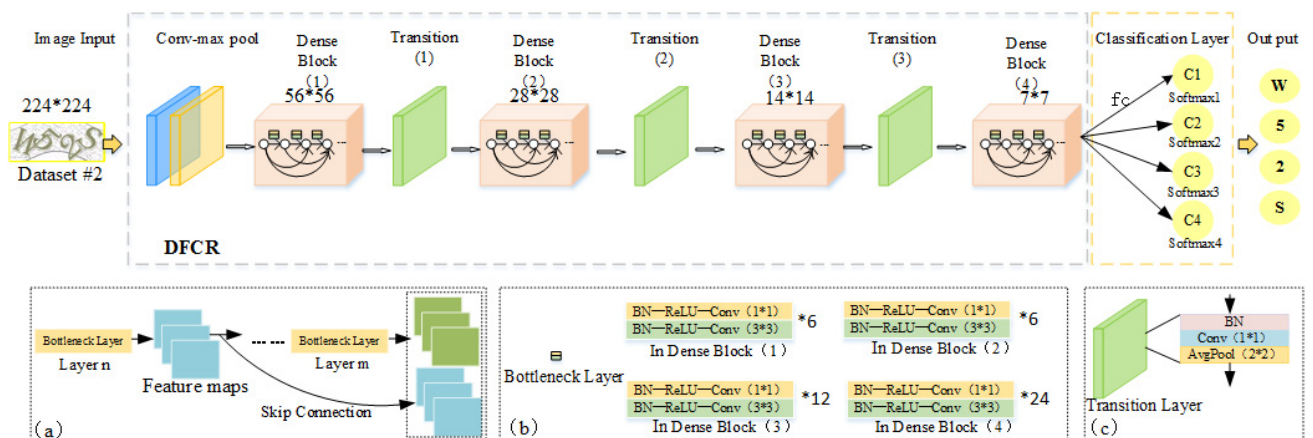
After that, 4 dense blocks are concatenated in turn. In each dense block, the "skip connection" and $\text{BN} \rightarrow \text{ReLU} \rightarrow \text{Conv}(1 \times 1) \rightarrow \text{BN} \rightarrow \text{ReLU} \rightarrow \text{Conv}(3 \times 3)$ are performed between every two layers, and the transition layer is connected after the first three dense blocks. The structure of the transition layer is constructed by $\text{BN} \rightarrow \text{Conv}(1 \times 1) \rightarrow \text{AvgPool}(2 \times 2)$ to implement down-sampling, which is used here to reduce the dimension of the feature-maps and parameters and helps eliminate the computational bottleneck. More importantly, we set the convolutional group of the bottleneck layers as 6 in the Dense Block 2, which is hugely different from Gao Huang's DenseNets.

Finally, the feature-maps are used to represent the confidence map of a class directly. The values in each feature-maps are added to obtain the average value, which is then taken as the confidence value of a class and input into the corresponding softmax layer for classification. The classification layer is composed of global average pooling and softmax, which has fewer parameters and effectively prevents data overfitting. Since the Dataset #1 has 5 characters, we use the multi-task classification method to access 5 softmax classifiers. The Dataset #2 has 4 characters, so the last fully connected layer is changed to 4 dense layers. For the Dataset #3 which has 4 characters, we only need to recognize the Chinese character that is rotated 90° at random, so the original network design can be maintained. The DFCR's architecture is shown in Table 1. The growth rate k is 32. Note that each "conv" layer shown in the table corresponds to the sequence $\text{BN} \rightarrow \text{ReLU} \rightarrow \text{Conv}$.

Figure 2 shows the process of identifying the image "W52S" by the DFCR we built. First, the network can be input 224×224 images directly. Then, it is linked with a convolution and max pool layer, and 4 dense blocks with 3 transition blocks, and produces 7×7 feature-maps. Particularly, we design 4 softmax layers at the end of the network for Dataset #2. (a) illustrates the "skip connection" is that the n^{th} layer is directly connected to the m^{th} layer. The n^{th} layer outputs k_1 feature-maps, and the m^{th} layer convolutes to get k_2 feature-maps, so the m^{th} layer outputs $(k_1 + k_2)$ feature-maps. (b) illustrates each dense block has a different number of bottleneck layers. (c) shows that the average pool layer is used to modified transition block.

Table 1. The architecture of DFCR.

Layers	Output Size	Dataset #1	Dataset #2	Dataset #3
Convolution	112×112		7×7conv, stride2	
Pooling	56×56		3×3max pool, stride2	
Dense Block (1)	56×56		$\left(\frac{1 \times 1conv}{3 \times 3conv}\right) \times 6$	
Transition (1)	56×56 28×28		1×1conv 2×2average pool, stride2	
Dense Block (2)	28×28		$\left(\frac{1 \times 1conv}{3 \times 3conv}\right) \times 6$	
Transition (2)	28×28 14×14		1×1conv 2×2average pool, stride2	
Dense Block (3)	14×14		$\left(\frac{1 \times 1conv}{3 \times 3conv}\right) \times 24$	
Transition (3)	14×14 7×7		1×1conv 2×2average pool, stride2	
Dense Block (4)	7×7		$\left(\frac{1 \times 1conv}{3 \times 3conv}\right) \times 16$	
Classification Layer	1×1	5×1000Dfully- connected, softmax	7×7global average pool 4×1000Dfully- connected, softmax	1000Dfully- connected, softmax







**Figure 2.** The architecture of the proposed DFCR.

4. Experimental results and analysis

4.1. Data sets

In this paper, we used three types of CAPTCHA images given by the organizing committee in the 9th China University Student Service Outsourcing Innovation and Entrepreneurship Competition, with each class consists of 15000 CAPTCHA images. We randomly selected 8000 for training, 2000 for validation and 5000 for test. The characteristics of the three types of CAPTCHA images are as follows: The Dataset #1 is a five-character CAPTCHA composed of 10 digits and 26 upper and lower case English letters randomly without slant. The Dataset #2 is a four-character CAPTCHA consisting of 10 digits and 26 uppercase English letters randomly, with skew, noisy and irregular curves. The Dataset #3 is a four-character CAPTCHA composed of Chinese characters, and one character is randomly rotated by 90°. So, the recognition difficulty of these three types of CAPTCHA images increases successively. Table 2 shows the CAPTCHA examples of the three types.

Table 2. Examples of CAPTCHA types.

Type	Sample 1	Sample 2
Dataset #1		
Dataset #2		
Dataset #3		

4.2. Experimental environment and parameter settings

We used the Windows 10 operating system, Inter(R) Core(TM) i5-8400 processor, GTX 1060, and our experiments were completed on Keras. Keras is a high-level neural network API which is very modular, minimal, and extensible.

In our experiment, we first normalized the CAPTCHA images to the size of 224×224 and converted it to the TFrecord format. All the networks were trained using stochastic gradient descent (SGD) with the initial learning rate $\alpha = 0.001$. Limited by the GPU running memory, we set the batch size as 16 for 100 epochs.

4.3. Comparison and analysis of experimental results

4.3.1. Training accuracy and loss value

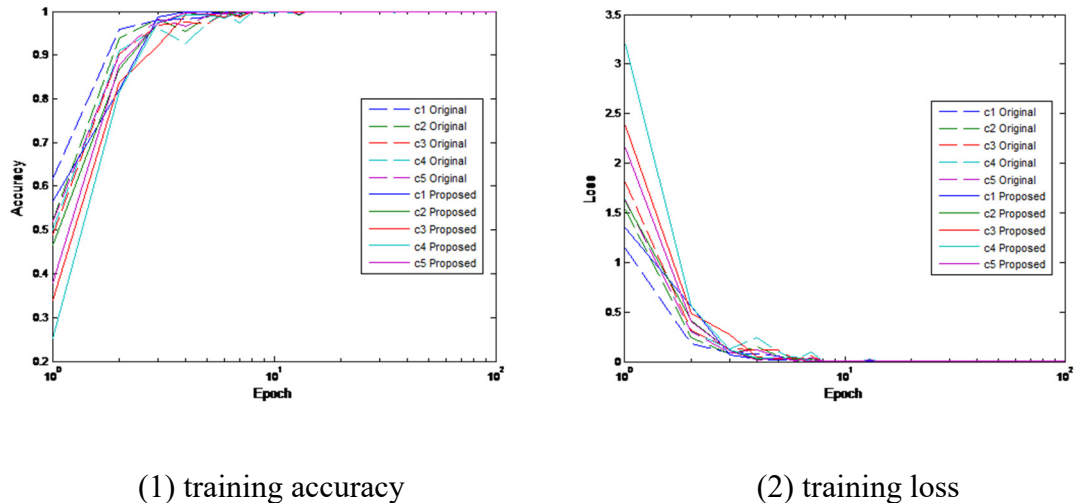


Figure3. Comparison of training accuracy and loss on the first type of CAPTCHA.

As shown in Figure 3, (1) and (2) respectively show the training accuracy and loss value of the Dataset #1 in 100 epochs, the solid line indicates 5 classifiers of DFCR, and the dotted line indicates DenseNet-121's. It can be seen from the exact value that although the training accuracy of DFCR is not as high as that of DenseNet-121 at begin, but the gradient of training accuracy is faster than DenseNet-121 in the subsequent iteration, especially in the 3 epoch. Then the exact value has reached 98.6%, which also shows that reducing the convolutional group is beneficial to improve the ability of the model to train and accelerate the convergence. At the same time, the DFCR loss value converges faster in the first three epochs. After 6 epochs, the model tends to be stable and the DenseNet-121 loss value converges more quickly in the first 6 epochs. After 11 epochs, the model is almost stable.

4.3.2. Memory consumption and training time

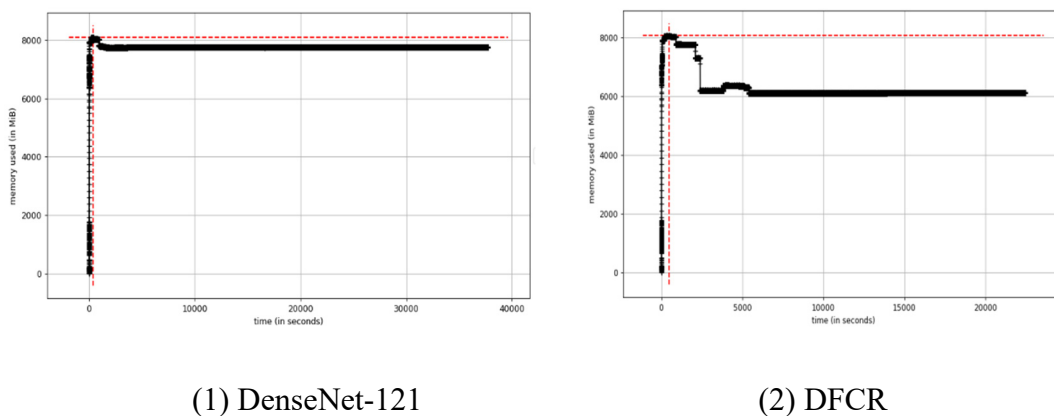


Figure4. Comparison of memory used and training time on the first type of CAPTCHA.

As shown in Figure 4, it is the memory consumption and the training duration for 100 epochs. It can be seen that the memory consumption of the DenseNet-121 network during training is close to 80%, and the DFCR we built is only about 60%. And the training time has been reduced by nearly 3 hours. Thus, the DFCR reduces memory consumption and model training time.

4.3.3. Identification accuracy and network parameters

We compare the CAPTCHA identification accuracy and parameters of DFCR with the ResNet-50 and the DenseNet-121. Three types of 5000 CAPTCHA test sets of the TFrecords format are input to the trained optimal model, then the recognition accuracy is performed according to the existing tags, and the results are recorded in Table 3 and 4.

Table 3. The identification accuracy of each network.

	Dataset #1		Dataset #2		Dataset #3	
	Validation set	Test set	Validation set	Test set	Validation set	Test set
	2000	5000	2000	5000	2000	5000
ResNet50	99.70%	95.34%	99.95%	99.90%	99.95%	99.86%
DenseNet-121	99.80%	95.40%	99.95%	99.90%	100%	99.92%
DFCR	99.80%	99.60%	100%	99.96%	100%	99.94%

Table 4. Comparison of each network parameters.

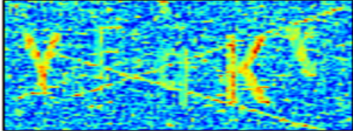
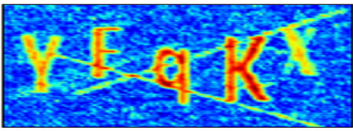
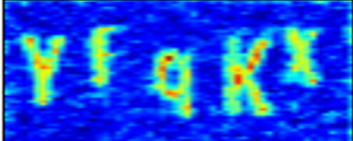
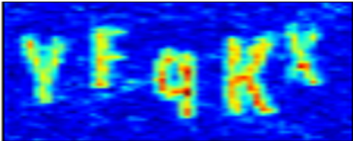
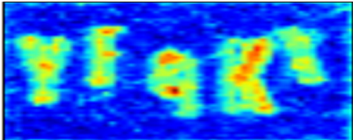
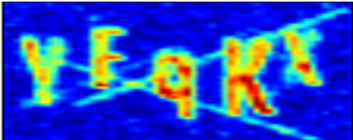
Parameters	Dataset #1			Dataset #2			Dataset #3		
	Total	Depth	Dimen	Total	Depth	Dimen	Total	Depth	Dimen
	Params		sion	Params		sion	Params		sion
ResNet50	23966777	177	2048	23890964	177	2048	23595908	177	2048
DenseNet121	7227129	428	1024	7189204	428	1024	7041604	428	1024
DFCR	3781833	302	784	3752788	302	784	5919940	386	976

As shown in Table 3 and 4, we can see that the DFCR have better recognition accuracy than the fine-tuning ResNet-50 and DenseNet-121. Especially on Dataset #1, the accuracy of the DFCR is 4.2% higher than DenseNet-121. Not only that, the total parameters and the feature dimensions of the ResNet-50 are several times than ours, which adds much difficulty to subsequent data processing. We cut the total number of parameters of DFCR to half of DenseNet-121's. Not only the dimension of the feature map is reduced, but the overall training time is reduced by several hours. It can be seen that it is not a mechanically deepening of the network to have an excellent classification effect. In practical applications, a neural network needs to be constructed for specific data.

4.3.4. Feature visualization

As shown in Table 5, we visualize the training process of CAPTCHA image "YEqKX." Specifically, we reconstruct the features of each layer of convolution and output a fixed feature. Even if the input of the same picture has a degree of transformation, the output can remain unchanged, which also indicates that the CNNs have strong robustness.

Table 5. Feature visualization of the DFCR compared with the DenseNet-121.

Layers	DenseNet-121	DFCR
conv1/relu		
conv2_block4_1_relu		
pool2_relu		

We visualize and superimpose the feature maps of the conv1/relu, conv2_block4_1_relu, and pool2_relu layers in each channel to obtain a visualization as shown in Table 5. Compared with the DenseNet-121, the DFCR we built has a stronger representation of the output characteristics in the same layer. In particular, in the output of the pool2_relu layer, it can be seen that the feature profile of the DFCR is more concrete than the DenseNet-121.

5. Discussion and conclusion

Although there are various kinds of CAPTCHAs, text-based CAPTCHA is applied most widely. On the one hand, it is because its a convenient and user-friendly way for website user; on the other hand, CAPTCHAs are a low-cost solution for websites. However, we know that the text CAPTCHAs are vulnerable and not as secure as expected. So we are willing to design text CAPTCHAs with higher security and better usability.

Defeating the CAPTCHAs is the most effective way to increase its own safety by finding the deficiency. The deep CNNs act as a more robust and useful method. All in all, using deep learning techniques to enhance the security of CAPTCHAs is a promising direction. In this paper, we constructed a deep CNN, which we referred to as DFCR. We compared its effectiveness with the ResNet-50 and the DenseNet-121. The experimental results showed that the DFCR not only kept compelling advantages but also encouraged feature reuse. On the one hand, memory consumption was greatly reduced. On the other hand, it had a better recognition performance than others. We used the end-to-end learning to directly identify the CAPTCHAs from the pixel image, which greatly avoided manual intervention, reduced the complexity of model training, and effectively prevented data over-fitting. It was different from traditional methods. What's more, we found that the recognition difficulty of these CAPTCHA images increases successively. So we can design CAPTCHA images by rotating multiple Chinese characters. The question of whether other CAPTCHA alternatives are robust and

whether the designs of new CAPTCHAs can be secure are still open problems and are part of our ongoing work.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61772561), the Key Research & Development Plan of Hunan Province (No.2018NK2012), the Science Research Projects of Hunan Provincial Education Department (No.18A174,18C0262), the Science & Technology Innovation Platform and Talent Plan of Hunan Province (No.2017TP1022).

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. L. Wang, R. Zhang and D. Yin, Image verification code identification of hyphen, *Comput. Eng. Appl.*, **28** (2011), 150–153.
2. J. Yan and A. S. E. Ahmad, A low-cost attack on a Microsoft CAPTCHA, *Proceedings of the ACM Conference on Computer and Communications Security*, (2008), 543–554.
3. L. Zhang, S. W. Huang, Z. X. Shi, et al., CAPTCHA recognition method based on LSTM RNN, *Pattern Recogn.*, **1** (2011), 40–47.
4. L. Yin, D. Yin and R. Zhang, A recognition method of twisted and pasted character verification code, *Pattern Recogn.*, **3** (2014), 235–241.
5. H. Li, J. H. Qin and X. Y. Xiang, An efficient image matching algorithm based on adaptive threshold and RANSAC, *IEEE Access*, **6** (2018), 66963–66971.
6. L.Y. Xiang, Y. Li and W. Hao, Reversible natural language watermarking using synonym substitution and arithmetic coding, *Comput. Mater. Con.*, **3** (2018), 541–559.
7. L.Y. Xiang, X. B. Shen, J. H. Qin, et al., Discrete multi-graph hashing for large-scale visual search, *Neur. Process. Lett.*, **49** (2019), 1055–1069.
8. Y. L. Liu, H. Peng and J. Wang, Verifiable diversity ranking search over encrypted outsourced Data, *Comput. Mater. Con.*, **1** (2018), 37–57.
9. H. T. Tang, *Verification code recognition model and algorithm of self-organizing incremental neural network*, MA thesis, Guangdong University of technology, 2016.
10. Y. Wang, Y. Q. Xu and Y. B. Peng, Verification code identification of xiaonei network based on KNN technology, *Comput. Moder.*, **2** (2017),93–97.
11. Y. S. Chen and Y. Zhang, Design and implementation of character-based image verification code recognition algorithm, *Comput. K. T.*, **1** (2017),190–192.
12. Y. Wang and M. Lu, A self-adaptive algorithm to defeat text-based CAPTCHA, *IEEE International Conference on Industrial Technology*, (2016), 720–725.
13. W. T. Ma, J. H. Qin, X. Y. Xiang, et al., Adaptive median filtering algorithm based on divide and conquer and its application in CAPTCHA recognition, *Comput. Mater. Con.*, **58** (2019), 665–677.
14. J. W. Wang, T. Li and X. Y. Luo, Identifying computer generated images based on quaternion central moments in color quaternion wavelet domain, *IEEE T. Circ. Syst. Vid.*, **1** (2018), 1.

15. X. W. Liu, L. Wang, Jian Zhang, et al., Global and local structure preservation for feature selection, *IEEE T. Neur. Net. Lear.*, **25** (2014), 1083–1095.
16. J. H. Qin, H. Li, X. Y. Xiang, et al., An encrypted image retrieval method based on Harris corner optimization and LSH in cloud computing, *IEEE Access*, **17** (2019), 24626–24633.
17. M. L. Wen, X. Zhao, M. Q. Cai, et al., End-to-end verification code recognition based on deep learning, *Wireless Inter. technol.*, **14** (2017), 85–86.
18. Y. Peng, Research on verification code recognition based on deep convolutional neural network, *Commu. world*, **1** (2018), 66–67.
19. Z. Zhang, S. F. Wang and L. Dong, Verification code recognition based on deep learning, *J. hubei univ. technol.*, **2** (2018), 5–11.
20. S. R. Zhou, W. L. Liang, J. G. Li, et al., Improved VGG model for road traffic sign recognition, *Comput. Mat. Con.*, **1** (2018), 11–24.
21. W. Fang, F. H. Zhang and V. S. Sheng, A method for improving CNN-based image recognition using DCGAN, *Comput. Mat. Con.*, **1** (2018), 167–178.
22. Y. P. Lv, F. P. Cai, D. Z. Lin, et al., Chinese character CAPTCHA recognition based on convolution-neural network, *Proceedings of the IEEE Congress on Evolutionary Computation*, (2016), 4854–4859.
23. G. Garg and C. Pollett, Neural network CAPTCHA crackers, *Proceedings of the Future Technologies Conference*, (2016), 853–861.
24. Y. H. Shen, R. G. Ji and D. L. Cao, Hacking Chinese touclick CAPTCHA by multiscale corner structure model with fast pattern matching, *Proceedings of the ACM International Conference on Multimedia*, (2014), 853–856.
25. W. Fan, J. G. Han, Fan Gou, et al., Chinese character verification code recognition by convolutional neural network, *Comput. Eng. Appl.*, **3** (2018), 160–165.
26. G. Huang, Z. Liu, L. V. D Maaten, et al., Densely connected convolutional networks, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (2017), 2261–2269.
27. N. Ma, X. Zhang, H. T. Zheng, et al., ShuffleNet V2: practical guidelines for efficient CNN architecture design, *Computer Vision and Pattern Recognition*, preprint, arXiv:1807.11164.



AIMS Press

©2019 the Author(s), licensee AIMS Press. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0>)