

POSITION-ENCODING CONVOLUTIONAL NETWORK TO SOLVING CONNECTED TEXT CAPTCHA

Ke Qing*, Rong Zhang

*Department of Electronic Engineering and Information Science
University of Science and Technology of China
No. 443 Huangshan Rd, Hefei, Anhui Province, 230027 P. R. China*

**E-mail: zrong@ustc.edu.cn*

Submitted: 6th October 2021; Accepted: 12th October 2021

Abstract

Text-based CAPTCHA is a convenient and effective safety mechanism that has been widely deployed across websites. The efficient end-to-end models of scene text recognition consisting of CNN and attention-based RNN show limited performance in solving text-based CAPTCHAs. In contrast with the street view image and document, the character sequence in CAPTCHA is non-semantic. The RNN loses its ability to learn the semantic context and only implicitly encodes the relative position of extracted features. Meanwhile, the security features, which prevent characters from segmentation and recognition, extensively increase the complexity of CAPTCHAs. The performance of this model is sensitive to different CAPTCHA schemes. In this paper, we analyze the properties of the text-based CAPTCHA and accordingly consider solving it as a highly position-relative character sequence recognition task. We propose a network named PosConv to leverage the position information in the character sequence without RNN. PosConv uses a novel padding strategy and modified convolution, explicitly encoding the relative position into the local features of characters. This mechanism of PosConv makes the extracted features from CAPTCHAs more informative and robust. We validate PosConv on six text-based CAPTCHA schemes, and it achieves state-of-the-art or competitive recognition accuracy with significantly fewer parameters and faster convergence speed.

Keywords: deep neural network, position encoding CNN, text-based CAPTCHA recognition, character recognition

1 Introduction

CAPTCHA, automatically identifying machine programs and human users [18], is an effective mechanism to block the operations from bots. With the development of automatic scripting techniques, most websites have to use this security mechanism to resist malicious attacks or resource abuse such as brute-force attacks, bulk registration, and automatic voting. While a wide variety of CAPTCHA types have been proposed, the text-based CAPTCHA is

the predominant scheme considering the superior usability and flexibility [21]. Correspondingly, the recognition of text-based CAPTCHA remains a significant research topic for it helps to obsolete the vulnerable CAPTCHA schemes and ensures that the deployed schemes are effective and can be trustworthy.

In CAPTCHAs, the character sequences are designed to be segmentation-resistant. The strokes of characters are connected or even overlapped, and the complicated background with dot and line noise

blurs the boundary of characters. Since support vector machine and multi-layer perceptron can accurately recognize the single character once it is taken apart from the sequence [4], the previous researches focus on the segmentation algorithms [9, 20, 13, 8]. However, the methods based on the two-stage framework consisting of segmentation and recognition are sensitive to the background texture and shape of characters. Despite effectiveness for certain schemes, the performance falls sharply when the CAPTCHA on the target website is updated.

Recently, many works have been proposed to simultaneously localize and recognize text in natural scene images within one-stage [19, 14, 12]. Compared with the two-stage framework, the end-to-end neural networks avoid intermediate processes, including preprocessing, segmentation, and postprocessing. The architecture typically consists of a convolutional neural network and recurrent neural network, in which CNN is used as an encoder to extract features from the input image and the following RNN decodes the features to the prediction sequence.

The method used to recognize scene text is also applied to solve connected text-based CAPTCHAs. An end-to-end model based on CNN and attention-based RNN is proposed to attack real-world text CAPTCHAs [23]. However, this network structure that is proven efficient in scene text recognition is still limited in solving CAPTCHAs.

Firstly, this hierarchical architecture is relatively large, in which the number of learnable parameters approaches tens of millions. To ensure the generalization of the model, a large volume of manually-labeled real CAPTCHAs is required to be collected. In addition, the embedded security features, such as character overlapping, distortion, rotation, and noisy interference, immensely expand the diversity of character texture. Consequently, the performance of the prior model applied in CAPTCHA recognition is sensitive to different schemes. To improve the accuracy of CAPTCHA recognition, the extracted features are required to be more informative and robust.

In this paper, we analyze the property of the character of text-based CAPTCHA and consider solving CAPTCHA as a highly position-relative and non-semantic character sequence recognition

task. For the semantic absence, we hypothesize that the RNN following the CNN in the end-to-end network only learns the relative position of features and compensates for the loss of position information in classic convolutional layers. To verify the hypothesizes, we propose a novel architecture, called a position-encoding convolutional (PosConv) network. In PosConv, we remove the RNN and merge the local features of characters with the relative position using a modified convolutional mechanism. The overall architecture of PosConv is simple and compact, containing only 1/8 parameters compared with the prior network [23].

PosConv applies a novel padding strategy in CNN layers to enhance the position-relative information in extracted features. During the convolutional computation on the feature maps, the local perceptive fields are padded with position-relative matrices in parallel, and correspondingly the convolutional kernel is expanded to fit the same size. Further mathematical derivation proves that the proposed padding mechanism can be implemented with two asymmetric operations which make encoded position information adaptive and learnable.






We tested PosConv on 6 text-based CAPTCHAs including a public available CAPTCHA scheme from the python library and 5 real schemes on popular websites (Baidu, NetEase, Sina, eBay, and Sohu). These schemes are sophisticated and cover diverse security features. Experimental results show that PosConv achieves state-of-the-art or competitive performance on all the schemes with significantly fewer labeled CAPTCHAs required in the training stage.

The rest of the paper is organized as follows. Section 2 introduces the properties of text-based CAPTCHAs and the prior methods to solve CAPTCHAs. Section 3 presents the proposed position encoding methods and the PosConv network is detailed in Section 4. In Section 5, we evaluate the PosConv on different CAPTCHA schemes and analyze the performance. Section 6 concludes the paper.

2 Background

Text-based CAPTCHA is an important security mechanism to protect websites from malicious at-

Table 1. CAPTCHAs on real popular websites embedded with different security features.

| CAPTCHA Scheme | Image Sample | Anti-recognition | | | | Anti-segmentation | | |
|----------------|---|------------------|-------|------------|----------|-------------------|-------|---------|
| | | fonts | sizes | distortion | rotation | background | noise | overlap |
| Ebay |  | | | ✓ | | | | ✓ |
| Baidu |  | ✓ | ✓ | | | | | ✓ |
| Ifeng |  | | | | ✓ | ✓ | ✓ | ✓ |
| Sina |  | ✓ | ✓ | | ✓ | | ✓ | ✓ |
| Souhu |  | | ✓ | | ✓ | ✓ | ✓ | |

tacks. In this section, we will introduce the evolving properties of the text-based schemes, and the prior methods proposed to solve them.

2.1 Security Feature

Early schemes of text-based CAPTCHAs simply grouped a few characters and are easily recognized. To tell the automatic script and the human apart, security features are introduced to increase the complexity and variability of the character sequence [1]. The function of security features can be divided into anti-recognition and anti-segmentation.

Anti-recognition features are used to make the shape of characters distorted. The characters in CAPTCHAs vary in fonts, sizes, and thickness. Besides, the image processing methods, such as affine transformation and blurring, are also employed to make the shape of characters fickle [13].

Anti-segmentation features prevent the solving methods from splitting the character sequence. The CAPTCHAs are usually placed on a complicated background, and the interval between characters is reduced. As a result, the strokes of characters are connected or even overlapped, blurring the boundary of characters in the sequence [7]. The CAPTCHA schemes embedded with different security features are shown in Table 1.

Researches have demonstrated that the classification methods can accurately recognize the separate character [4]. Therefore, the anti-segmentation feature is primary in text-based schemes to render the schemes not vulnerable to automatic solving methods.

2.2 Prior Solving Methods

The prior methods can be divided into two-stage and end-to-end frameworks.

Two-stage methods consist of segmentation and recognition. The performance of these methods relies largely on the correctness of segmentation. These methods observe certain schemes on pixel-level and then craft the corresponding specific segmentation algorithms. A method based on color filling and projection was proposed to handle the character sequence associated with consecutive noise [20]. Edge and fuzzy logic segmentation was applied to overlapped characters without dot noise [13], and the Log-Gabor filter was used to split slightly connected characters with deformation and rotation [9]. For Microsoft's two-layer CAPTCHA, the researchers presented a mixed segmentation method based on the professional analysis of the security feature: a customized algorithm to split the hollow and solid characters and another to split the upper and lower lines [8]. Two-stage methods

shared the advantage that considerable performance can be reached with small datasets. However, the application range of the specific methods is relatively narrow due to the limited generalization.

In contrast to the two-stage framework that considers segmentation and recognition as two distinct tasks, the one-stage framework solves the CAPTCHA in a single step. In prior works, the end-to-end networks were proposed to take the CAPTCHA image as the input instead of segmented characters and predict the label sequence directly.

Researchers proposed a unified approach to recognizing arbitrary multi-digit numbers from Street View imagery via a deep CNN [6]. The best performance is achieved by the deepest architecture that has 11 hidden layers. With this model, they use millions of CAPTCHA images as the training set and reach 99.8% accuracy on transcribing the reCAPTCHA scheme. To solve CAPTHCAs requiring fewer labeled images, a model based on a generative adversarial network was proposed. In this method, a synthesis is trained to produce CAPTHCAs similar to the target images, and a pre-processing model is trained to remove the security features. Note that a pre-defined parameter setting is necessary to constrain the security feature space covering the target CAPTCHA schemes [22]. The end-to-end model which has been proven effective in street view imagery was also utilized to recognize CAPTHCAs. A network consists of CNN and attention-based RNN achieves relatively higher performance on CAPTHCAs deployed by 11 websites [23]. However, this deep and hierarchical structure still contains tens of millions of learnable parameters. To avoid overfitting, a large volume of manually-labeled real CAPTHCAs is required to feed the model. Meanwhile, as the security features are diverse, the performance of this model is sensitive to different CAPTCHA schemes.

3 Method

In this section, we demonstrate the property of character sequence in CAPTCHA and consider its recognition as a highly position-relative task. To make the extracted features more informative and robust, a position encoding method is proposed to leverage the position information of the character sequence.

3.1 Position-Relative Property of Character Sequence in CAPTCHA

Different from the text in document and street view, the character sequence in CAPTCHA is non-semantic. For the text-based CAPTCHA, effective contextual information exists where the characters are connected or overlapped. The connected-stroke has a significant impact on the original shape and leads to difficulty in recognition.

Some confusing examples and the corresponding prediction are shown in Table 2. The ground truth of the character is listed in the first column, and the CAPTCHA samples are listed in the second column. In the top row, the sequence of characters is labeled as ‘J’, ‘A’, ‘N’, and ‘R’. For this example, the character ‘N’ is sensitive to connected strokes and easily recognized as ‘W’ or ‘M’. The probability of the false prediction is relative to the position that the character occurs. Intuitively, the probability that ‘N’ is mistaken as ‘W’ descends when the character occurs at the left-most position, and the probability mistaken as ‘M’ descends at the right-most position. Similarly, the vertical position of ‘N’ is relative to the robustness of the character and a lower position reduces the probability of both mistakes. Meanwhile, some pairs of characters easily cause connected-stroke confusion. The pair of ‘U’ and ‘J’ are recognized as ‘W’ in the second row. The character ‘Y’ is mistaken as ‘I’ for ‘C’ on the left absorbs its stroke in the third row.

Based on the examples of connected-stroke confusion, we consider the recognition of text-based CAPTCHA as a highly position-relative task. The position-relative property of the character sequence, such as the absolute of the sensitive character and the relative position of easily-confusing pairs, should be leveraged to solve CAPTHCAs.

3.2 Position Encoding Method

The concept of position encoding is introduced in natural language processing tasks [17]. Because all words of the input sequence are fed to the network with no special order or position, position-relative information is embedded in each word to help the model incorporate the order of words.

Position encoding mechanism based on Convolutional Neural Networks (CNNs) has also been studied in prior works. In image recognition tasks,

Table 2. Examples of connected-stroke confusion.

| char | Unstable position | preds | Robust position | preds |
|------|-------------------|-----------|-----------------|-----------|
| N | | N: 0.3501 | | N: 0.9539 |
| | | W: 0.6015 | | W: 0.0076 |
| J | | J: 0.3015 | | J: 0.9707 |
| | | V: 0.6388 | | V: 0.0102 |
| y | | y: 0.1926 | | Y: 0.9302 |
| | | l: 0.7281 | | l: 0.0184 |
| C | | C: 0.3818 | | C: 0.9837 |
| | | Q: 0.4295 | | Q: 0.0028 |

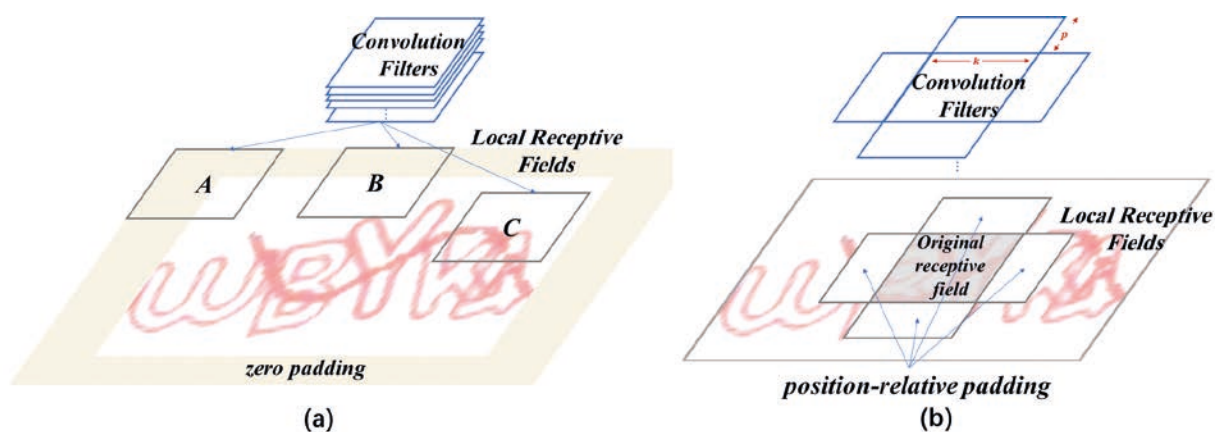


Figure 1. The padding methods of classic CNN and PosConv.

CNN extracts features with convolutional filters but ignoring the corresponding position [15]. The classic CNN model is considered to be spatially agnostic. However, recent work demonstrates that CNN indeed learns to encode position information implicitly [10]. This property of CNN mainly relies on the zero-padding strategy that is widely adopted in convolutional operations. The results show that the model with larger kernel size and deeper layers captures more position information.

The classic padding method is illustrated in Figure 1 (a). The zero-padding is implemented on the borders as an anchor from which position information is derived and eventually propagated to each extracted feature as the filter slides across the entire image. However, as the proportion of involved zero-padding varies with the distance to the border, the encoded position information is inconsistent among the local receptive fields. In Figure 1 (a), field A locating on the corner encodes significantly richer position information than interior field C via zero-padding. This position encoding is implicit due to the underlying propagation of position information. For field C, the extracted feature involves no position information via zero-padding in the current convolutional layer, and the position information is obtained through propagation from the fields that locate nearer to the border.

The proposed network PosConv in this paper is inspired by the conclusion that zero-padding introduces the position information in CNN. Different from the implicit position encoding, PosConv encodes position information into the extracted features explicitly with a novel padding strategy and modified convolutional operation. As illustrated in Figure 1 (b), we firstly expand the convolution filters along with the horizontal and vertical directions, respectively. Then we pad the local receptive fields synchronously to match the shape of expanded filters. Instead of zero-padding, we use the position of the padding block as the value. The convolution filter in PosConv consists of the central part and the expanded part. The central part extracts the features from the previous layer similar to the classic convolutional kernel. The size of the central part is $k \times k$ and the padding size along the horizontal and vertical direction is denoted as p .

3.3 Convolution in PosConv

PosConv uses asymmetric convolutions with shape $(k + 2p) \times 1$ and $1 \times (k + 2p)$ as:

$$\begin{aligned} y_{i,j}^{(l+1)} &= \sum_{m=i-s}^{i-s+k+2p-1} W_{m-i-s}^v \cdot x_{m,j}^{(l)} + b_{m-i-s}^v \\ z_{i,j}^{(l+1)} &= \sum_{n=i-s}^{i-s+k+2p-1} W_{n-i-s}^h \cdot y_{i,n}^{(l+1)} + b_{n-i-s}^h \\ x_{i,j}^{(l+1)} &= f(\text{bn}(z_{i,j}^{(l+1)})), \end{aligned} \quad (1)$$

where W^v and W^h are the asymmetric convolutional kernels. b^v and b^h are the corresponding bias. $x^{(l)}$ denotes the padded local receptive field of layer l . Batch normalization and ReLU active function are applied to yield $x^{(l+1)}$ as the input of next layer. The subscript i, j of the variables x, y , and z denote the coordinate in the feature maps. Note that when convolution filters slice across the feature maps with stride s , the position-relative padding may fall outside the image. We assign the padding value as -1.0 on the left/top, and 1.0 on the right/bottom.

The prior work [10] demonstrated that the encoded position information increases with the size of zero-padding. In PosConv, the padding value is set as coordinate value divided by the size of the corresponding dimension, the asymmetric convolution along the horizontal direction can be written as:

$$\begin{aligned} S_{exp} &= \sum_{m=l+1}^{l+k} W_{m-l}^c \cdot x_m + \sum_{m=1}^p W_m^L \cdot x_{l-m} \\ &+ \sum_{m=1}^p W_m^R \cdot x_{l+k+m} + b, \end{aligned} \quad (2)$$

where x denotes the vector of features with length $k + 2p$ in the expanded receptive field. l is the beginning index of the convolved receptive field. The first term denotes the weighted sum of the original receptive field. The following terms denote the convolution on the padding blocks. According to the distribution of padding values, S_{exp} can be rewritten as:

$$\begin{aligned} S_{exp} &= S_{org} + l \cdot \frac{p}{D} \sum_{m=1}^p (W_m^L + W_m^R) + \frac{p \cdot k}{D} \sum_{m=1}^p W_m^R \\ &+ \frac{1}{D} \sum_{m=1}^p m \cdot (W_m^R - W_m^L) + b. \end{aligned} \quad (3)$$

The convolution in the original receptive field is replaced as S_{org} . As p, D , and k are constant, W_m^L

and W_m^R are the vectors of trainable parameters. Eq. 3 can be written as:

$$S_{exp} = S_{org} + \frac{l}{D}W_{pos} + b_{pos}, \quad (4)$$

where

$$W_{pos} = p \sum_{m=1}^p (W_m^L + W_m^R)$$

$$b_{pos} = b + \frac{p \cdot k}{D} \sum_{m=1}^p W_m^R + \frac{1}{D} \sum_{m=1}^p m \cdot (W_m^R - W_m^L). \quad (5)$$

PosConv takes W_{pos} as a single trainable parameter to learn how much the position information should be encoded to the receptive fields. Instead of assigning the padding size p as a hyper-parameter and the weights W_m^L and W_m^R as trainable parameter vectors, we optimize the position-relative W_{pos} and b_{pos} via backpropagation directly. In this method, the position information encoded into the extracted features is adaptive, and few extra parameters are involved.

4 Network Architecture

The architecture of PosConv is briefly illustrated in Figure 2. The main difference between PosConv and the aforementioned end-to-end model is the removal of RNN for the semantic absence in the character sequence of CAPTCHA. Instead, PosConv encodes the position information via the modified convolution, not only leveraging the relative position between characters more efficiently and compactly but significantly reducing the number of parameters compared with the prior hierarchical model. Another difference is the convolutional kernel size used in PosConv is set as small as 1×3 and 3×1 . In contrast with the large kernel size used to capture more position information in classic CNNs [10], we can accelerate computation and further reduce the number of parameters.

In Figure 2, PosConv takes the CAPTCHA image and its position map as the input. The asymmetric convolutions are performed simultaneously on the feature maps and the corresponding position map. The 1×3 and $1 \times 2p$ convolutional kernels slice across the feature maps and the position map synchronously. We encode the position information into the extracted as the sum illustrated in Eq. 4 for

every receptive field. The max-pooling and ReLU activation are uniformly performed on the sum of one-dimensional convolutions.

In the PosConv architecture, the modified units, which we refer to as position-encoding convolutional layers, are stacked to extract more informative and robust features from the CAPTCHA images. The output is flattened and then fed to the separate fully-connected layers corresponding to individual characters. Considering that the characters in the CAPTCHA sequence are uniformly generated and subject to the same distribution, we share the weights of all the fully-connected layers in the network.

5 Experiment

To demonstrate the effectiveness we test PosConv on various connected text-based CAPTCHA schemes in this section. Firstly, we use a public CAPTCHA library to compare the model with the prior end-to-end models. Additionally, the effectiveness of the proposed position encoding convolution is analyzed based on this dataset. Then, we use PosConv to solve five complicated text-based CAPTCHA schemes deployed on real websites, evaluating our method more comprehensively.

5.1 Performance Evaluation of PosConv

In this section, PosConv is evaluated on a connected text-based CAPTCHA scheme, which is a public library named Captcha 0.2.4 as shown in Table 3. We use this library as the dataset for the reason that real CAPTCHA schemes collected in prior works are seldom released considering the security of target websites. Meanwhile, the CAPTCHAs individually collected by researchers cannot be guaranteed consistent, and some schemes used in prior work have been deprecated for the frequent website update.

In this experiment, we evaluate the performance of models as the depth increases. The channels of convolutional layers are [32, 32, 64, 80, 96, 128, 192, 256] respectively. The kernel size is fixed as 3×3 in each layer. Batch normalization is implemented before ReLU activation. The shapes of the following separate fully-connected layers and soft-

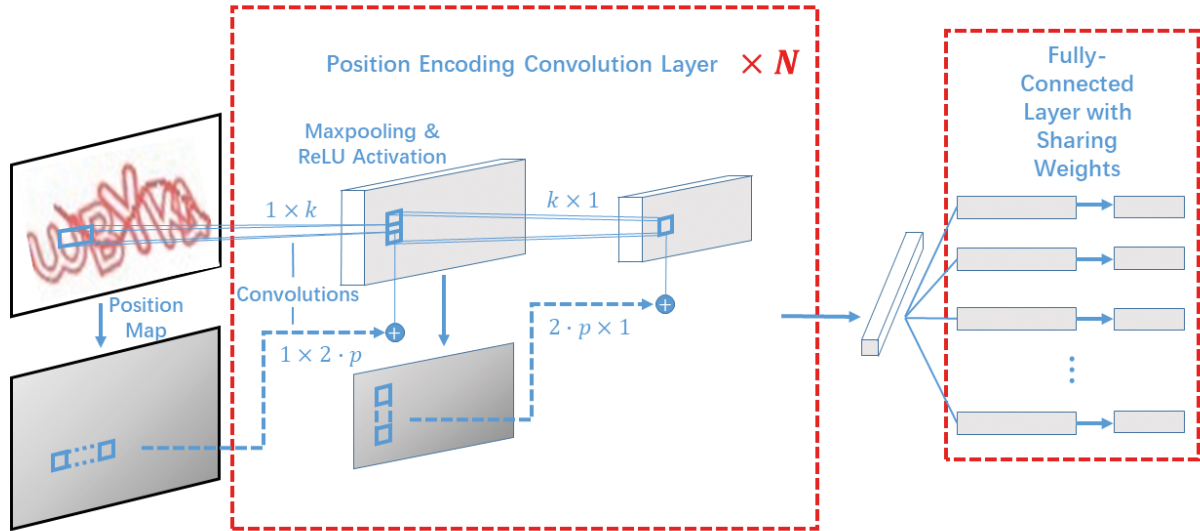


Figure 2. The overview architecture of PosConv.

Table 3. Schemes of connected text-based CAPTCHAs used for evaluation.

| Scheme | sample | size | length | Char categories |
|--------------------------------|--------|--------|--------|-----------------|
| Captcha 0.2.4 (public library) | | - | 4 | 26 |
| sina | | 100000 | 4 | 28 |
| ebay | | 20000 | 6 | 10 |
| souhu | | 20000 | 4 | 30 |
| baidu | | 10000 | 4 | 25 |
| netease | | 10000 | 4 | 30 |

max layer are the same. We use Adam as the gradient descent optimization algorithm with an initial learning rate of 0.001, and a custom decay scheduler is used in which the rate reduces to [0.0005, 0.0001, 0.00005, 0.00001] according to the decrease of the loss.

The base in Figure 3 denotes the inception-v3 model, which is used as the backbone of all the models. We use character recognition accuracy to measure the performance and the result shows that PosConv outperforms the attention-based RNN model for all the depth of networks. The difference is distinct when the network is relatively shallow. The performance of the attention-based RNN model is limited until the number of layers reaches 4 while PosConv achieves approximately 90% recognition accuracy when the network contains only 2 stacked convolutional layers. This demonstrates that the position-encoding convolution is efficient and extracts more informative and robust features.

Note that the depth of network only represents the number of convolutional layers in attention-based RNN model, and the following LSTM decoder is not involved. By contrast, PosConv has the relatively smaller model size and significantly fewer parameters, yielding advantages in solving real CAPTCHAs described in Section 5.4 later.

5.2 The Effect of Position Encoding

In this Section, we compare PosConv with the prior position encoding methods based on CNN including PosENet [10] and CoordConv [11]. PosENet is the aforementioned network padding the image on the border. We set the padding size as 2 which achieves the optimal performance [10]. The CoordConv extends standard convolution with extra channels which are filled with (constant, untrained) coordinate information [11]. Note that the channel for r coordinate is used in this experiment. As before, all the models use inception-v3 as the backbone and share the same hyper-parameters.

The performance of different position-encoding methods are shown in Figure 3. The character accuracy of PosConv improves smoothly and stably as the depth increases, obviously superior to PosENet and CoordConv. Meanwhile, Figure 4 shows that the loss of PosConv reduces rapidly, and the training epochs before convergence are less. After 75

epochs, PosConv achieves the performance superior to all other well-trained models. Note that at the very beginning of training, the loss of PosConv reduces at a relatively slow rate. This is probably because PosConv learns the direction of gradient descent of the introduced trainable parameters W_{pos} and b_{pos} on the initial stage. Then the loss reduces rapidly in approximately 5-10 epochs and the performance surpasses other models, proving the effectiveness of proposed position encoding convolutional mechanism.

5.3 Attack Real CAPTCHAs

To evaluate PosConv more comprehensively, we collect 5 CAPTCHA schemes from real websites, including Sina, eBay, Sohu, Baidu, and NetEase. These schemes cover various complicated security features and are representative of connected text-based CAPTCHAs. We collect the images with a python script and use a data annotation service to manually label them.

For each scheme, 1000 samples are used for validation, 500 samples for testing, and the rest for training. We use sequence accuracy as the evaluation of the performance in attacking real CAPTCHAs, requiring the model to recognize all the characters in the sequence correctly simultaneously.

Different from the Captcha 0.2.4 library, the datasets collected from real websites are limited in data size. We set the L2 regularization penalty as 0.001 and label smoothing as 0.9 to regularize the model. Meanwhile, we use the following data augmentation: firstly, we randomly crop the image to 0.9 ratios and then rotate the image ranging from -30° to 30° . Finally, we resize it to the initial size with bilinear interpolation.

The sequence accuracy of PosConv and prior CAPTCHA solvers are listed in Table 4. Note that some schemes collected in prior work are deprecated due to frequent website updates. Only the schemes remaining consistent and publicly accessible are involved. For the given schemes, PosConv achieves better or at least competitive sequence accuracy. The average recognition accuracy is above 90%, which means PosConv solves all the schemes approximately in a single try. The performance significantly exceeds the widely accepted standard

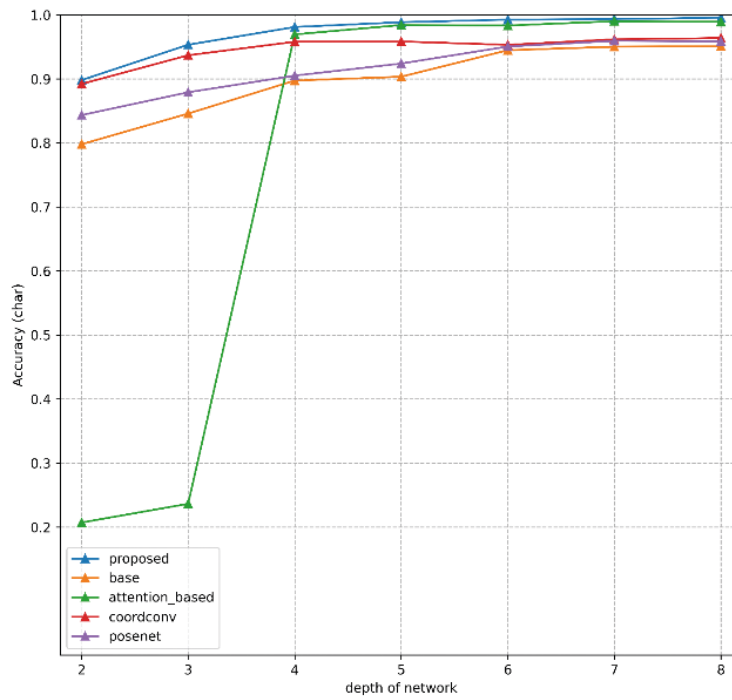
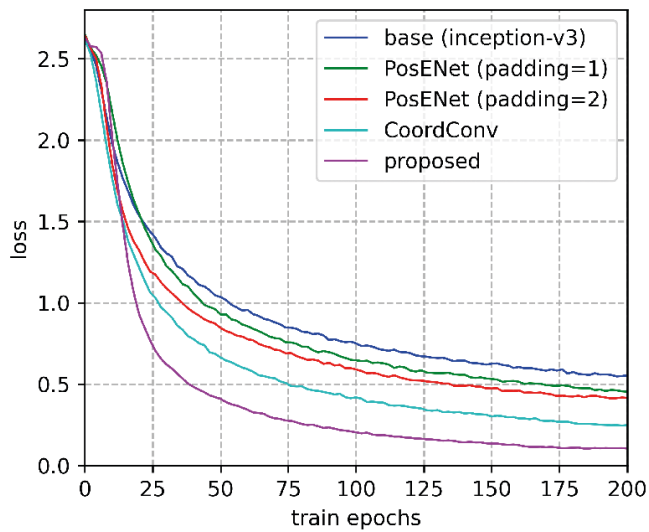


Figure 3. The character accuracy of models on solving text-based CAPTCHAs



| Model | Min loss | Accuracy (char) | Early stop (epochs) |
|---------------------|----------|-----------------|---------------------|
| base (inception-v3) | 0.5229 | 0.8991 | 210 |
| PosENet (padding=1) | 0.4716 | 0.9267 | 234 |
| PosENet (padding=2) | 0.4480 | 0.9310 | 208 |
| CoordConv | 0.2531 | 0.9718 | 244 |
| Proposed | 0.0952 | 0.9962 | 201 |

Figure 4. The loss curves of models based on different position encoding methods.

Table 4. The sequence accuracy of PosConv and prior CAPTCHA solvers.

| Website | Baidu | Sina | Ebay | Souhu | Netease |
|------------------|-------|-------|-------|-------|---------|
| Proposed model | 96.0% | 98.2% | 97.2% | 90.8% | 92.6% |
| Gao | 44.2% | 9.4% | 58.8% | - | - |
| Bursztein et al. | - | - | 51.2% | - | - |
| Zi | 95.8% | 86.0% | - | - | - |
| Tang et al. | 57.0% | 75.0% | - | - | - |
| Hussain et al. | - | - | 27.0% | - | - |
| Liu et al. | - | - | - | 67.7% | - |
| Duan et al. | 96.8% | 96.8% | 97.3% | - | 90% |

for CAPTCHA security that one scheme is broken when the attacker is able to reach a precision of at least 1% [3].

5.4 Analysis of Solving Methods

In this Section we analyze the text-based CAPTCHA recognition methods and explain the advantages of PosConv in solving CAPTCHAs from real websites.

The method proposed by Gao is based on a hard-coded segmentation algorithm at pixel level [9], achieving 44.2% and 58.8% accuracy on Baidu and eBay, respectively. This scheme-specific attack is sensitive to the texture of background and the shape of characters. For the Sina scheme, the accuracy drops rapidly to 9.4%. Similarly, the methods of Bursztein [2] and Tang [16] are based on segmentation, which are relatively vulnerable to distortion and deformation.

The end-to-end models take the CAPTCHA image as the input and predict the character sequence directly, not requiring expert-designed segmentation. Faster R-CNN used in Duan’s work achieves a competitive accuracy in solving various schemes of CAPTCHAs [5]. This work collected a fine-annotated CAPTCHA dataset in which the characters are manually marked with a bounding box. Compared with the weakly supervised learning methods, the method based on faster R-CNN is labor-intensive and time-consuming. Zi used an end-to-end model based on CNN and attention-based RNN to attack Google CAPTCHA

[23], and the success rate is 87.9% and 98.3% with 50,000 and 200,000 training samples, respectively. In this network, CNN is used to extract local features, and the attention-based RNN is used to learn the position information of the non-semantic character sequence. This model is further evaluated on CAPTCHA schemes deployed by 11 websites, achieving the accuracy ranging from 74.8% to 97.3%. The PosConv proposed in our work removes the RNN in the end-to-end model and uses a novel position encoding convolution to extract the informative and robust features. Compared with Zi’s network containing 8.46 million parameters, PosConv significantly reduces the number to 0.83 million.

6 Conclusion

In this paper, we propose an end-to-end model named PosConv to solve connected text-based CAPTCHAs. Considering the CAPTCHA recognition as a highly position-relative and non-semantic character sequence recognition task, we remove the RNN used in prior end-to-end models for scene text recognition. Instead, we propose a position encoding convolution in which a novel padding strategy for local receptive fields is implemented to leverage the position information in the character sequence. We validate PosConv on six text-based CAPTCHA schemes and it outperforms state-of-the-art methods and achieves competitive recognition accuracy with significantly fewer parameters and faster convergence speed.

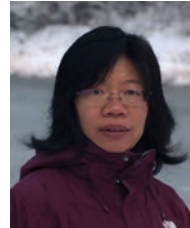
References

- [1] Darko Brodić, Alessia Amelio, Nadeem Ahmad, and Syed Khuram Shahzad. Usability analysis of the image and interactive captcha via prediction of the response time. In *International Workshop on Multi-disciplinary Trends in Artificial Intelligence*, pages 252–265. Springer, 2017.
- [2] Elie Bursztein, Jonathan Aigrain, Angelika Moscicki, and John C Mitchell. The end is nigh: Generic solving of text-based captchas. In *8th {USENIX} Workshop on Offensive Technologies ({WOOT} 14)*, 2014.
- [3] Elie Bursztein, Matthieu Martin, and John Mitchell. Text-based captcha strengths and weaknesses. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 125–138, 2011.
- [4] Kumar Chellapilla, Kevin Larson, Patrice Y Simard, and Mary Czerwinski. Computers beat humans at single character recognition in reading based human interaction proofs (hips). In *Conference on Email and Anti-Spam (CEAS)*, pages 1–8, 2005.
- [5] Chen Duan, Rong Zhang, and Ke Qing. Feature refine network for text-based captcha recognition. In *International Conference on Image and Graphics*, pages 64–73. Springer, 2019.
- [6] Ian J. Goodfellow and Yaroslav Bulatov and Julian Ibarz and Sacha Arnoud and Vinay Shet, Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks, 1312.6082, 2014.
- [7] Ahmad Salah El Ahmad, Jeff Yan, and Lindsay Marshall. The robustness of a new captcha. In *Proceedings of the Third European Workshop on System Security*, pages 36–41, 2010.
- [8] Haichang Gao, Mengyun Tang, Yi Liu, Ping Zhang, and Xiyang Liu. Research on the security of microsoft’s two-layer captcha. *IEEE Transactions on Information Forensics and Security*, 12(7):1671–1685, 2017.
- [9] Haichang Gao, Jeff Yan, Fang Cao, Zhengya Zhang, Lei Lei, Mengyun Tang, Ping Zhang, Xin Zhou, Xuqin Wang, and Jiawei Li. A simple generic attack on text captchas. In *The Network and Distributed System Security Symposium (NDSS)*, pages 1–14, 2016.
- [10] Md Amirul Islam, Sen Jia, and Neil D. B. Bruce. How much position information do convolutional neural networks encode?, 2020.
- [11] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution, 2018.
- [12] Pengyuan Lyu, Minghui Liao, Cong Yao, Wenhao Wu, and Xiang Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–83, 2018.
- [13] Rabih Al Nachar, Elie Inaty, Patrick J Bonnin, and Yasser Alayli. Breaking down captcha using edge corners and fuzzy logic segmentation/recognition technique. *Security and Communication Networks*, 8(18):3995–4012, 2015.
- [14] Liang Qiao, Ying Chen, Zhanzhan Cheng, Yunlu Xu, Yi Niu, Shiliang Pu, and Fei Wu. Mango: A mask attention guided one-stage scene text spotter, 2020.
- [15] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules, 2017.
- [16] Mengyun Tang, Haichang Gao, Yang Zhang, Yi Liu, Ping Zhang, and Ping Wang. Research on deep learning techniques in breaking text-based captchas and designing image-based captcha. *IEEE Transactions on Information Forensics and Security*, 13(10):2522–2537, 2018.
- [17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [18] Luis Von Ahn, Manuel Blum, and John Langford. Telling humans and computers apart automatically. *Communications of the ACM*, 47(2):56–60, 2004.
- [19] Zbigniew Wojna, Alexander N Gorban, Darshyang Lee, Kevin Murphy, Qian Yu, Yeqing Li, and Julian Ibarz. Attention-based extraction of structured information from street view imagery. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, pages 844–850. IEEE, 2017.
- [20] Jeff Yan and Ahmad Salah El Ahmad. A low-cost attack on a microsoft captcha. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 543–554, 2008.
- [21] Guixin Ye, Zhanyong Tang, Dingyi Fang, Zhanxing Zhu, Yansong Feng, Pengfei Xu, Xiaojiang Chen, Jungong Han, and Zheng Wang. Using generative adversarial networks to break and protect text captchas. *ACM Transactions on Privacy and Security (TOPS)*, 23(2):1–29, 2020.

- [22] Guixin Ye, Zhanyong Tang, Dingyi Fang, Zhanxing Zhu, Yansong Feng, Pengfei Xu, Xiaojiang Chen, and Zheng Wang. Yet another text captcha solver: A generative adversarial network based approach. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, pages 332–348, 2018.
- [23] Yang Zi, Haichang Gao, Zhouhang Cheng, and Yi Liu. An end-to-end attack on text captchas. IEEE Transactions on Information Forensics and Security, 15:753–766, 2019.



Ke Qing is a PhD candidate in the Department of Electronic Engineering and Information Science, University of Science and Technology of China. His main research areas are OCR and CAPTCHA recognition.



Rong Zhang has a Ph.D. in signal and information processing. She is currently an associate professor in the Department of Electronic Engineering and Information Science, University of Science and Technology of China. Her main research areas are digital image processing, remote sensing image processing, and data compression.