SEMI-SUPERVISED APPROACH TO TRAIN CAPTCHA LETTER POSITION DETETOR

Ondrej Bostik
Doctoral Degree Programme (5), FEEC BUT
E-mail: bostik@feec.vutbr.cz

Supervised by: Karel Horak
E-mail: horak@feec.vutbr.cz

Abstract:

Common Optical Character Recognition (OCR) methods benefit from the fact, that the text is distributed in images in a predictable pattern. This is not the situation with CAPTCHA systems. Utilizing OCR algorithms to overcome common web anti-abuse CAPTCHA systems is therefore a challenging task. To train a system to overcome any CAPTCHA scheme, an attacker needs a huge dataset of annotated images. And for some methods, the attacker needs not only the right answers but also an exact position of the character in the CAPTCHA image.

Annotate the positions of the object in an image is a time-consuming task. In this paper, we propose a system, which can help to annotate the position of CAPTCHA character with minimal human interaction. After annotating a small sample of targeted CAPTCHA images, a YOLO-based region detection deep network is used to search for the characters' locations.

Keywords: OCR, CAPTCHA, Deep learning, YOLO v2, semi-supervised learning, MATLAB

1 INTRODUCTION

Nothing is most synonymous with an anti-abuse system used on the world-wide-net nowadays as a CAPTCHA (Completely Automated Public Turing Test to tell Computers and Humans Apart). And the major part of the commonly used CAPTCHA system world-wide is based on OCR (Optical Character Recognition) problem.

An image with characters scattered around is sent as a part of almost every web form. Every user who wants automatically interact with this kind of service needs to create a system to overcome CAPTCHA protection [1, 2]. The side effect of creating new systems to overcome text-based CAPTCHAs is therefore an improvement in the field of OCR [3].

Recent CAPTCHA breaking systems are based on convolutional neural networks (CNN). One great example is [4], one deep network is used for the length of the text estimation and the other deep network is using this information for classification estimation. Another similar work [4] utilized CNN for feature extraction and Long Short-Term Memory (LSTM) for recognition. In both cases, a huge annotated dataset is needed for the successful attack.

The work presented in [5] overcomes some of these issues by transfer learning. The initial solver is created with the use of synthetic CAPTCHA images. After that 500 manually annotated CAPTCHA images are used for fine-tuning this solver. Although the base solver has an average success rate close to 0%, after fine-tuning on real examples of targeted CAPTCHA images the success rates increases to 50%-97%.

2 PROPOSED SOLUTION

The work described in this paper is based on our previous work in [6]. In our precedent paper, we propose using the target web-page for generating a sufficient enough dataset without annotating a huge number of data. To create a versatile end-to-end system, we need a huge training dataset containing not only the right answer (which we can obtain by the algorithm presented in [6]) but also the correct positions of the characters in every CAPTCHA image.

Annotate the positions of every character in the CAPTCHA image is a time-consuming task, so we want to automate this task by utilizing a deep learning approach. Every CAPTCHA image is processed by region detector based on YOLO v2 (You Only Look Once) [7, 8] algorithm with deep network based on ResNet 50 architecture in a role of feature extraction unit.

The last part of the proposed solution is the filtering stage. Because the CAPTCHA challenges are generated automatically without human interaction, each implementation takes into account the possibility that the presented CAPTCHA image will not be human-readable. The user can request an alternate image without penalization. Therefore, it is more advantageous for the CAPTCHA breaking algorithm to decide that the image is unreadable and request a new image instead of sending a bad response.

In future work, the automatic validation stage will follow. The extracted region will be fed into the classification stage, where every region is recognized via classic OCR methods (for example another deep network, Tesseract OCR engine, or for example simple k-NN algorithm). If the answer is correct for all the characters, we can save the image with the correct annotation. This method relies on the previously known transcription of the CAPTCHA text, which can be extracted by the methods from [6], or simple human transcription. Either way, this kind of annotation is far quicker, then forcing the human to hand-labelling every character in the image.

In this paper, we substitute the validation engine with a human for greater control over the annotation results.

2.1 INPUT DATASET OVERVIEW

For this experiment, we enlarged the dataset used in our previous work [6]. With the use of the same python web crawler utilizing Selenium Webdriver we automatically obtained 50 000 CAPTCHA images from the domain `http://geocheck.org`. For this experiment, we annotated 250 CAPTCHA images for training the object detector and we also extract 1 000 more unannotated images for validation. The targeted CAPTCHA scheme consists only of digits with a fixed length of 5 numbers.

The main advantage of this CAPTCHA scheme, in particular, is the verification process, which is located on the client-side. The correct answer is presented in the HTML code sent to the end-user computer in the form of an MD5 hash, which can be broken rapidly utilizing the simple rainbow table. Our automated system can download and break around 4 CAPTCHA images per second exploiting this weakness. Assembling the huge dataset of real-world CAPTCHAs all annotated with 100% correct answers is then a simple task. The sample images from the used dataset are displayed in figure 1.

Figure 1: Sample of input dataset used for the experiments

2.2 IMPLEMENTATION

For the rapid deployment, the experiment was performed in MATLAB computational environment using Deep Learning Toolbox and Computer Vision Toolbox. Proposed experimental setup is visualized on figure 2.

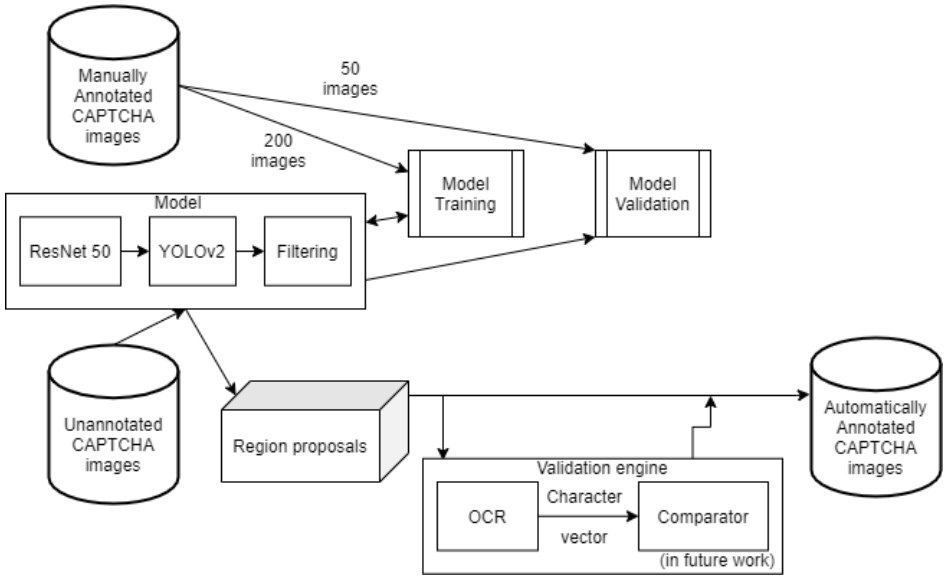

Figure 2: Experiment setup overview (including future work)

Before the experiment started, we annotated 250 CAPTCHAs images via *Image Labeler MATLAB app* to create precise ground-truth data of character location.

On every epoch during the learning stage, the input dataset was augmented with random transformations. Every image undergoes random changes, in contrast, hue, saturation, and brightness, after that the image was randomly scaled with the cropping to preserve the original size.

Images with annotated regions were used to train segmentation deep neural network [9] based on ResNet [10] as a feature extraction segment and YOLO v2 [7, 8] as region proposal segment with layers shown on table 1. Training used Adam Stochastic Optimization algorithm [11], initial learning rate set to 0.001 and mini-batch size of 16 for the total number of 25, 50, or 100 epochs. The size of anchor boxes for YOLO layers was estimated according to [8], the number of anchor boxes was set to 7.

	Layer type	Layer description
1	Convolution	1024pcs 3x3x1024 convolutions with stride 1 and padding 'same'
2	Batch normalisation	Batch normalization with 1024 channels
3	ReLU	ReLU
4	Convolution	1024pcs 3x3x1024 convolutions with stride 1 and padding 'same'
5	Batch normalisation	Batch normalization with 1024 channels
6	ReLU	ReLU
7	Convolution	42pcs 1x1x1024 convolutions with stride 1 and padding [0 0 0 0]
8	YOLO v2 Transform Layer	YOLO v2 Transform Layer with 7 anchors
9	YOLO v2 Output	YOLO v2 Output with 7 anchors

Table 1: YOLO based object detector layers

The last part of our model is the filtering stage. The YOLO object detection network for each image outputs a number of bounding boxes. As the evaluated schema contains a fixed number of digits, we can easily filter the results with the incorrect number of regions accordingly to our expectations. The second filtering stage is based on identifying the overlapping segments. For each pair of bounding boxes in the image, the Intersection over Union (IoU) is calculated and when it reaches threshold experimentally set to 40% intersection, the regions are merged into one.

3 RESULTS OF EXPERIMENT

The presented experiment was realized on a laptop computer with Intel Core i5-6300HQ with 4 cores and 2.3GHz frequency and with graphic card NVIDIA GeForce GTX 950M. The presented segmentation network was trained on 200 manually annotated images.

Part of this experiment was evaluating the speed of training on common hardware. The training time per 25 epochs was 52 minutes and grows linearly with more training epochs, with the 100 epochs, the training time was almost 3.5 hours.

For validation 50 more CAPTCHA images were manually annotated. Using these images the validation accuracy for classifier trained for 100-epoch reached 95.90%. Resulting comparison of validation accuracy is depicted in figure 3.

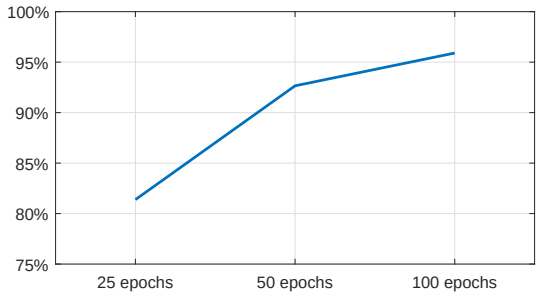

Figure 3: Validation accuracy per training epoch on manually labeled data

To further assess the experiment 1.000 unlabelled images were used. Without the filtering stage, there is a significant difference in the portion of images with the correct number of segments. Images with the correct number of segments were then manually evaluated by a human. After the filtering stage, there is only a marginal difference between 50-epoch and 100-epoch learning, both closed to the 100% region proposal accuracy. The results are summarized in table 2.

Training epochs	Training time	Validation accuracy	Portion of images with the correct number of region without / with filtering	Manually evaluated region proposal accuracy without / with filtering
25 epochs	52m	81.38%	27.60% / 79.10%	66.30% / 96.97%
50 epochs	1h 44m	92.66%	40.00% / 89.80%	91.50% / 99.55%
100 epochs	3h 25m	95.90%	77.10% / 94.50%	99.09% / 99.79%

Table 2: Segmenting process success rate per learning epochs

4 CONCLUSION

This paper presents an idea of a semi-supervised system for the position annotation of characters in the CAPTCHA image. In this paper, we present a system requiring only a very small initial dataset

used for classifier training. Utilizing 200 annotated images we achieved the accuracy of 99.09% of character region detection for 3.5hours of learning on a laptop computer without filtering. Even more, with the simple filtering presented in this paper, we can achieve 99.55% accuracy with a detector trained only for half the time.

The following work will focus on adding the validation engine based on OCR recognition of CAPTCHA images. Furthermore, this data can be used to quickly train end-to-end CAPTCHA breaking system without any human interaction.

ACKNOWLEDGEMENT

The completion of this paper was made possible by the grant No. FEKT-S-20-6205 - "Research in Automation, Cybernetics and Artificial Intelligence within Industry 4.0" financially supported by the Internal science fund of Brno University of Technology.

REFERENCES

[1] E. Bursztein, J. Aigrain, A. Moscicki, and J. C. Mitchell, "The End is Nigh: Generic Solving of Text-based CAPTCHAs," 2014.

[2] E. Bursztein, A. Moscicki, C. Fabry, S. Bethard, J. C. Mitchell, and D. Jurafasky, "Easy Does It: More Usable CAPTCHAs," in *CHI '14 Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, (1600 Amphitheatre Pkwy), pp. 2637–2646, 2014.

[3] K. Kaur and S. Behal, "Designing a Secure Text-based CAPTCHA," in *Procedia Comput. Sci.*, vol. 57, pp. 122–125, Elsevier, 2015.

[4] M. Tang, H. Gao, Y. Zhang, Y. Liu, P. Zhang, and P. Wang, "Research on Deep Learning Techniques in Breaking Text-Based Captchas and Designing Image-Based Captcha," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, pp. 2522–2537, oct 2018.

[5] P. Wang, H. Gao, Z. Shi, Z. Yuan, and J. Hu, "Simple and Easy: Transfer Learning-Based Attacks to Text CAPTCHA," *IEEE Access*, vol. 8, pp. 59044–59058, 2020.

[6] O. Boštík, "SEMI-SUPERVISED DEEP LEARNING APPROACH FOR BREAKING GEOCACHING CAPTCHAS," in *Proc. II 26th Conf. STUDENT EEICT 2020 - Sel. Pap.*, (Brno), pp. 166–170, Vysoké učené Technické, Fakulta elektrotechniky a komunikačních technologií, Vysoké učené Technické, Fakulta elektrotechniky a komunikačních technologií, 2020.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 779–788, 2016.

[8] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 6517–6525, 2017.

[9] K. Horak and R. Sablatnig, "Deep learning concepts and datasets for image recognition: overview 2019," in *Elev. Int. Conf. Digit. Image Process. (ICDIP 2019)*, no. 11179, pp. 484–491, SPIE, 2019.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 770–778, 2016.

[11] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *Int. Conf. Learn. Represent.*, 2014.