ATLANTIS PRESS

Advances in Engineering Research (AER), volume 132

2017 Global Conference on Mechanics and Civil Engineering (GCMCE 2017)

Container Code Detection by Deep Convolutional Network

WANG Zhi-Ming[*,a], MA Shu[b]

[*]School of Computer and Communication Engineering, University of Science and Technology Beijing, China

[a]wangzhiming@ies.ustb.edu.cn,[b]mashu1010@qq.com

Keyword: Container code recognition, Deep convolutional network, Holistic-nested edge detection.

Abstract. Automatic container code recognition plays an important role in customs logistics and tra-nsport management. Due to difficulties such as character color and font size variation, illumination conditions, image degradation, and exist of many other characters, automatic detect and recognition of container code is still a difficult task. This paper proposes a container code detection algorithm based on deep convolutional neural network named holistically-nested edge detection (HED). In the training phase, a bounding box was drawn around container code as virtual edges, and they were feed to the network together with original image to train the HED model. In the test phase, probability map of bounding box was predict by trained model and finally bounding box is obtained by thresholding and connected region analysis. Experimental results on 9953 container images show that the performance of IOU and recall precision on test set can reach 0.646 and 0.934 respectively with the proposed algorithms.

1. Introduction

As one of the most popular transporting carrier in international trade, container is widely used in docks and depots. Manual inspect and register container identity code is slowly, tedious and prone to make mistakes. It is necessary to extract and recognize container identity automatically by image process and pattern recognition techniques. Though automatic container code recognition had been studied since 1990's [1, 2], there were still many difficulties in container code recognition, especially in code detection and char extraction. The main difficulties come from: (1) Container code characters were painted with a wide range of sizes, fonts, colors, and spacing. The characters of container code canbe aligned in one, two, three, or even four horizontal lines, and one or two vertical lines. (2) Different illumination conditions (sunny or cloudy, day or night) make foreground / background contrast quite different from image to image. (3) There are many noises among container code characters, such as rust, mud, peeling paint, and fading color. (4) There are also many other objects around container code, such as company logos, container weight characters, rail bars, etc. Some of typical container code images are shown in Fig. 1.

In recent years, deep learning has achieved great success in various computer vision and pattern recognition areas. Many deep learning based object detection algorithms give excellent results, such as RCNN[3], fast RCNN[4], SSD[5], faster RCNN[6] , YOLO[7], Yes-Net[8], and RON[9]. But these algorithms were designed to detect a entire object with connected regions. However, container code can hardly be considered as one object, as show in Fig. 1. It is more likely to be a special kind of object which is composed by several small elements, which makes container code detection quite different with other object detection.

In this paper, we proposed a container code detection based on a deep convolutional network named holistically-nested edge detection (HED)[10]. In the training phase, virtual edges were draw around container code to be detected. Then these image and label pairs were used to train a deep neural network. In the test phase, edges were detected from an input image, and this edge map was threshold and segmented to obtain the final container code region.

Fig. 1. Typical Container ID Images

The rest of this paper is organized as follows: Section 2 review some recent container code recognition algorithms. In Section 3, the proposed algorithm and architecture of deep neural network were analyzed and described in detail. Section 4 gives extensive experimental results on a large container code databased from real-world scene. Finally, conclusions and some discussion were given in Section 5.

2. Related Works

From 1990, there have many studies on automatic container code extraction and recognition. Lee，extracts container code by four steps: remove background by mathematic morphology processing; eliminate noise by character size; locate characters by a 4-boundary detection; and find code by char grouping. But as the character size is quite different, the shape and size of the morphology Structuring Element (SE) is difficult to be decided. An extraction algorithm through gray level quantization, thresholding and stroke analysis was also proposed by Lee [11, 12]. But quantization may give different pixels different gray level in the same character when the gray level is nearby the quantization threshold, and this will result in many broken strokes. Further, as the contrast in container code image is quite different, it is difficult to give a suited quantize factor.

Mullot extracts container code by locate text lines first, with the knowledge that text have significant horizontal transitions with certain frequency [13]. But when the characters are slant or with few characters in one line, there won't be any evident peaks in the transition histogram. Igual extracts container code by top-hat morphology operations [14], Gu and Wang extracts container code by medium filter and top-hat morphology operations, followed ty thresholding and gray beep points analysis, but these algorithms also face the difficulty of suitable SE and threshold selection [15,16]. HE's method performs an vertical edge detection, followed by a morphology operation, find text lines by projection, and extract container code by template match method [17, 18]. His method also suffers from font size and noises in projection operation. Pan locates container code by count horizontal and vertical gray level alternations in specific size and frequency, similarly, font size changes and other objects make it difficult to success [19].

All above extraction methods are based on some morphology operations. These methods will fail when the font size changes in a wide range. This difficult may be solved by scan the image several times with different morphology SE. But this will increase the processing time evidently.

Chen first get candidate character region by medium filter, vertical edge detection and connected-region analysis, then find container code by a hidden Markov model [20]. But it strongly relies on prior knowledge of character size, interval space.

Wu use large vertical filter to detect character, and find text region by height and position filter, and

connect region analysis [21]. Then, container code characters were cut by binarization and projection. But it will be difficult if container code is arranged in vertical or spaces between characters are large enough that they cannot be considered as a text line. Yoon adopts appearance-based character detection algorithm, which finds characters on an input image by inspecting every region using a sliding window across the image [22]. However, it needs post process to find container code in all possible characters.

The proposed method treats all 11 characters in a container code as a holistic entirety. Our model is complete end-to-end trainable, and it is naturally multi scale detection. It needs no prior parametersettings such as font size, segmentation threshold, code arrangement manner, etc.

3. Container Detection by HED

As mentioned above, complicated arrangement of container code makes it quite different with other object detection problems. In order to train a deep neural network to detect container code as a holistic object, we apply a new edge detection algorithm named HED.

3.1 Data Preprocess.

HED algorithm was originally designed to detect multi scale edge in an image. The problem is that is no explicit edge around 11 container code characters. If we use the naive HED algorithm, we would find edges including characters and all other edges resulted by logo, rail bars, etc. In order to train an end-to-end network for container code detection, we manually draw a bounding box around true container and ignore all true edges in the image. Typical training data pairs were show in Fig. 2.

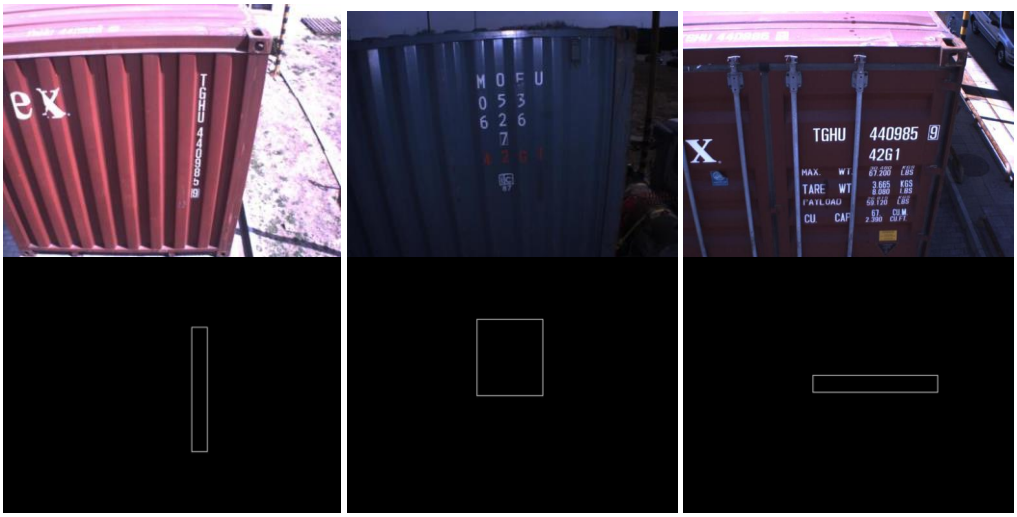

Fig. 2. Typical container code images (top) and bounding box label (bottom)

3.2 Framework of Detection

When an image becomes blurring, its high frequency energy will becomes lower, and the magnitude of wavelet detail coefficients of an image will becomes smaller. However, if these coefficients are normalized by their maximum and minimum value, coefficients of blur image will become more similar, or flat. Then the entropy computed from this normalized coefficient will becomes larger with image becomes blurring.

3.3 Post Processing

The output of HED is a probability map, indicates the probability of every pixel being on a container code bounding box. To get the finally prediction bounding box, we need to finish three post processing step: (1) Thresholding the out image to get a binary map. (2) Search the connected areas and keep the largest one. (3) Search the bounding box of the largest connect region as finally container code region. The flow of post processing is shown in Fig. 4.

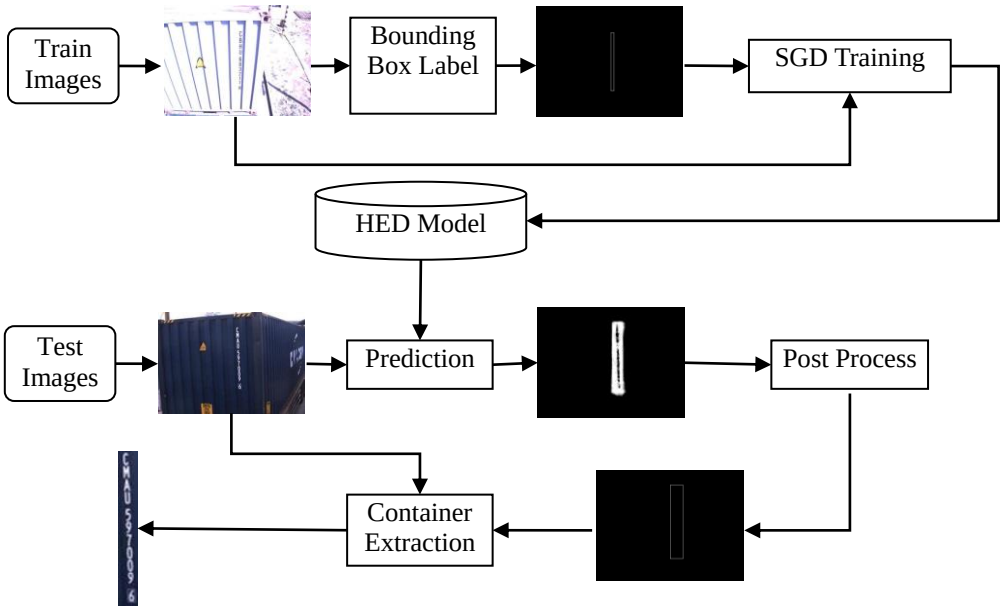

Fig. 3. Container code detection flow

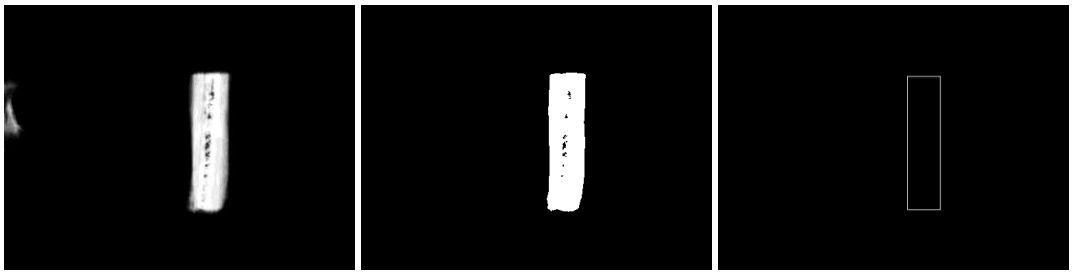

Fig. 4. Post processing (from left to right: initial predict result, threshold result, and bounding box)

4. Experimental Results

Our experimental data include 9,953 images captured from custom dock with various illumination conditions in day and night. The image resolution is 768*1024 pixels. All images were labeled with container code bounding box. Among them, 8,510 images were used for training HED algorithm, and 1,443 images were used for performance evaluation.

Typical detection results were shown in Fig. 5. Clearly, container code regions were detected precisely even with quite different font size and many disturbance of other characters, rail bar, and other noise.

We used two criterions to evaluate proposed container code detection algorithm, detection precision and recall ratio. Detection precision is defined by intersection over union (IOU), which is the ratio of overlap area between true container code region and predicted region to union area of them. Recall ratio is defined by ratio of overlap area to true code region.

$$IOU = \frac{P \cap T}{P \cup T}. \tag{1}$$

$$Recall = \frac{P \cap T}{T}. \tag{2}$$

Here P is the predicted container code region, and T is the true code region.

Experimental results on all of the train and test images were show table 1. The IOU is promising, and about 93.5% percent of container code were detected on test set.

Fig. 5. Typical container code images (top) and detection results (bottom)

Table 1. Container code detection result

Dataset	Sample Number	IOU	Recall
Train	8,510	0.745	0.995
Test	1,443	0.646	0.934

5. Conclusion

A novel container code detection algorithm based on HED is proposed in this paper. All 11 container code characters were treated as a holistic region no matter what pattern they were arranged in an image. After adequate train, true container can be detected while other characters were effectively neglected. Experimental result on large dataset is quite promising. Future researches include train other object detection algorithm, recognize and verify the container code.

References

[1] C. M. Lee, F. Gao, Segmentation algorithm in container ID number recognition system. *International Conference of Automation, Robotic and Computer Vision*, 1990.

[2] H. C. Lui, C. M. Lee, and F. Gao, Neural networks application to container number recognition. *IEEE 14th Annual International Computer Software and Applications Conference*, 1990.

[3] R. Girshick, J. Donahue, T. Darrell, *et al*, Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[4] R. Girshick, Fast R-CNN. *IEEE International Conference on Computer Vision*, 2015.

[5] W. Liu, D. Anguelov, D. Erhan, *et al*, SSD: Single Shot MultiBox Detector. *European Conference on Computer Vision*, 2015.

[6] S. Ren, K. He, R. Girshick, *et al*, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1167-1180, 2017.

[7] J. Redmon, S. Divvala, R. Girshick, *et al*, You Only Look Once: Unified, Real-Time Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[8] L. Ma, X. Kan, Q. Xiao, *et al*, Yes-Net: An effective Detector Based on Global Information. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[9] T. Kong, F. Sun, A. Yao, *et al*, RON: Reverse Connection with Objectness Prior Networks for Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[10] Z. Tu, S. Xie, Holistically-Nested Edge Detection. *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[11] C. M. Lee, A. Kankanhalli, Automatic extraction of characters in complex scenes image. *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 9, pp. 67-82, 1995.

[12] C. M. Lee, W. K. Wong and H. S. Fong, Automatic Character Recognition for Moving and Stationary Vehicles and Containers in Real-life Images. *International Joint Conference on Neural Networks*, 1999.

[13] R. Mullot, C. Olivier, J. L. Bourdon, *et al*, Automatic extraction methods of container identity number and registration plates of cars. *International Conference on Industrial Electronics, Control and Instrumentation*, 1991.

[14] I. S. Igual, G. A. García and A. P. Jiménez, Preprocessing and recognition of characters in container codes. *16th International Conference on Pattern Recognition*, 2002.

[15] S. Gu, X. Luo, An Efficient Algorithm of Container Code Characters Location. *2nd International Conference on Computer Engineering and Technology*, 2010.

[16] W. Wang, M. Xie, A novel location method of characters in container codes. 2010 *2nd International Conference on Education Technology and Computer*, 2010.

[17] Z. W. He, J. L. Liu, H. Q. Ma, *et al*, A New localization method for container auto-recognition system. *IEEE International Conference Neural Networks and Signal Processing*, 2003.

[18] Z. W. He, J. L. Liu, H. Q. Ma, *et al*, A new automatic extraction method of container identity codes. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 6, pp. 72-78, 2005.

[19] W. Pan, Y. S. Wang, and H. J. Yang, Robust container code recognition system. *Fifth World Congress on Intelligent Control and Automation*, 2004.

[20] M. Chen, W. Wu, X. Yang, *et al*, Hidden-Markov-model-based segmentation confidence applied to container code character extraction. *IEEE Transactions on Intelligent Transportation Systems*, 2011.

[21] W. Wu, Z. Liu, M. Chen, *et al*, An automated vision system for container-code recognition. *Expert Systems with Applications* Vol. 8, pp.2842-2855, 2012.

[22] Y. Yoon, K.-D. Ban, H. Yoon, *et al*, Automatic container code recognition from multiple views. *ETRI Journal*, Vol. 38, pp. 767-775, 2016.