Pattern Recognition Letters 159 (2022) 77–83

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

CE-text: A context-Aware and embedded text detector in natural scene images

Yirui Wu[a], Wen Zhang[a], Shaohua Wan[b,*]

[a] College of Computer and Information, Hohai University, Fochengxi Road, Nanjing 210093, China
[b] School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan 430073, China Department of Information Systems, King Abdulaziz University, Jeddah 21589, Saudi Arabia

A R T I C L E I N F O

Article history:
Received 3 March 2020
Revised 7 February 2022
Accepted 6 May 2022
Available online 8 May 2022

Edited by XXX

Keywords:
Context-aware and embedded text detector
Hierarchial attention scheme
Visual content analysis
Deep compression method

A B S T R A C T

With the significant power of deep learning architectures, researchers have made much progress on effectiveness and efficiency of text detection in the past few years. However, due to the lack of consideration of unique characteristics of text components, directly applying deep learning models to perform text detection task is prone to result in low accuracy, especially producing false positive detection results. To ease this problem, we propose a lightweight and context-aware deep convolutional neural network (CNN) named as CE-Text, which appropriately encodes multi-level channel attention information to construct discriminative feature map for accurate and efficient text detection. To fit with low computation resource of embedded systems, we further transform CE-Text into a lighter version with a frequency based deep CNN compression method, which expands applicable scenarios of CE-Text into variant embedded systems. Experiments on several popular datasets show that CE-Text not only has achieved accurate text detection results in scene images, but also could run with fast performance in embedded systems.

1. Introduction

Due to the large variations in text and complex backgrounds, many deep learning based techniques have been proposed to improve the accuracy and robustness of text detection. However, these methods greatly suffer from slow optimization and detection speed, since each individual component must be trained and parameter tuning separately. There exists a trend in directly predicting word bounding boxes through an lightweight and single neural network. For example, [1] uses a single fully-convolutional network coping with bounding boxes of extreme aspect ratios to perform fast text detection [2]. However, such methods generally regard text as one class of objects without involving unique text characteristics for higher accuracy. Moreover, their proposed method are only deployed and tested on systems with Nvidia Titan GPUs, due to their high requirement for computation resource.

With this context, we build a context-aware and embedded text detector named as CE-Text to help detect text in natural scene images with high accuracy and efficiency. Fig. 1 shows the workflow of CE-Text, where step (a) inputs a natural scene image, step (b) represents a hierarchical channel-wise attention scheme to generate text context-aware feature map for higher detection accuracy, a lightweight convolutional neural network builded on feature map is proposed in (c) to predict multiple bounding boxes and corresponding text existence probabilities, step (d) presents an embedded version of text detector constructed by a frequency-based CNN compressing method, and step (e) obtains text detection results with bounding boxes and associated probabilities. Our motivations to involve techniques mainly rely on the following two considerations:

1) To overcome factors for wrong predictions, it's essential to construct highly discriminative features for accurate detection. By stacking different layers, a CNN extracts image features through a hierarchical representation of visual abstractions. Therefore, features extracted from CNN structure are essentially channel-wise and multi-layer. However, not all the features are equally important and informative for detection of text components. Therefore, we propose a hierarchical attention scheme to encode text context-aware information, which automatically focuses on text-related characteristics and discriminative feature channels for accurate text detection.

2) The main difficulty to shift deep neural networks to embedded systems lies in their high request for computational and storage intensity. For example, original YOLO is over 230MB in size, containing 30 layers and $6.74 * 10^8$ parameters. Running YOLO with such a tremendous number of parameters consumes large storage and computational resources. Due to the nature of local pixel cor-

* Corresponding author.
E-mail address: shwanhust@zuel.edu.cn (S. Wan).

https://doi.org/10.1016/j.patrec.2022.05.004
0167-8655/© 2022 Elsevier B.V. All rights reserved.

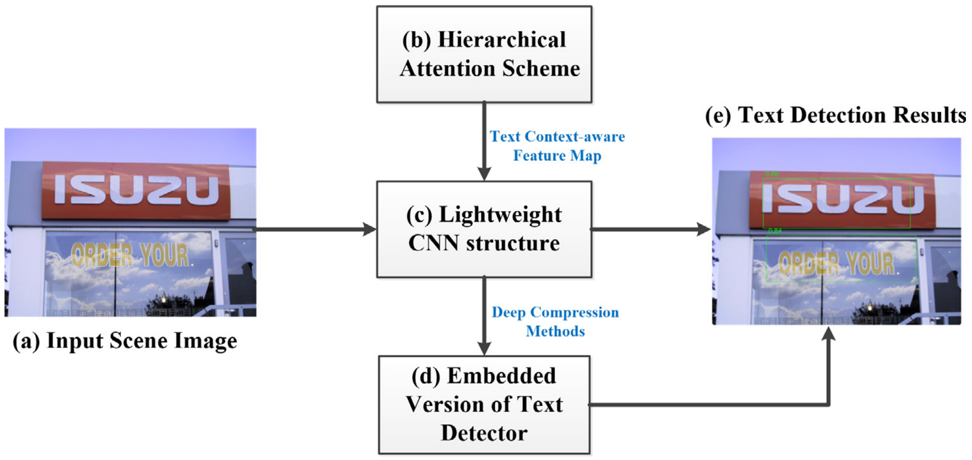
(a) Input Scene Image
(b) Hierarchical Attention Scheme
Text Context-aware Feature Map
(c) Lightweight CNN structure
Deep Compression Methods
(d) Embedded Version of Text Detector
(e) Text Detection Results
ISUZU
ORDER YOUR

Fig. 1. Workflow of the proposed CE-Text.

relation in images, *i.e.*, spatial locality, parameters of channel filters in the proposed text detectors or other visual content analysis systems tend to be smooth. Therefore, we convert parameters matrix into frequency domain and only utilize their low-frequency parts. The embedded version of CE-Text is thus appropriate to apply in embedding systems with low storage and memory size.

The contributions of this paper are three-fold:

- We propose a novel deep and context-aware CNN structure for text detection, in which a task-specified hierarchical attention scheme is adopted to enhance feature representative ability on the basis of multi-level and channel-wise context information.
- A hierarchically channel-wise attention scheme is carefully designed with channel-wise and multi-layer features, involving inherent and unique context characteristics of text components for better detection results.
- CE-Text adopts a novel frequency-based deep compression method to build a lightweight and embedded text detector, which have properties of highly-representative feature map, fast computing speed and small storage size.

2. Related work

2.1. Text detection methods

We generally category recent methods on text detection as regression based and segmentation based text detection [3]. Scene text detection have been greatly affected by the thought of directly predicting location without proposals and post-processing steps. EAST [4] propose a simple yet powerful pipeline that yields fast and accurate text detection for words or text lines of arbitrary orientations with a single neural network. SegLink [5] first predicts text segments and then links text segments to form text boxes by a single and SSD-style network, which has archived impressive results on long oriented text in natural scene.

Inspired by recent progress of fine-level image segmentation, some methods cast text detection as a semantic segmentation problem. For example, [6] adopt a modified FCN to produce multiple heatmaps corresponding to various properties of text, such as text region and orientation. To better separate adjacent text instances, [7] distinguish each pixel by deep neural networks into three categories: non-text, text border and text [8,9]. Textsnake [10] further describe a text instance as a sequence of ordered, overlapping disks centered at symmetric axes, each of which is associated with potentially variable radius and orientation and such geometry attributes are estimated via a Fully Convolutional Network (FCN) model. Most recently, [11] combine the ideas of the two types of methods, *i.e.*, regression and segmentation based text detection, while avoiding their shortcomings [12]. The idea of constructing natural scene graph is inspired by several works [13], which are designed to help object detection by bringing knowledge about scene layout or context information.

2.2. Deep compression methods

Deep neural networks have demonstrated the state-of-the-art power to complete classification or regression tasks. Based on compression time, we generally category deep compression methods into two groups: compressing during and after training. Compressing during training methods usually try to compress weights, activations and gradients during training, in order to obtain smaller and faster network. For example, [14] train binarized neural networks (BNNs) with binary weights and activations at run-time, which achieves a high compression result on multiple datasets.

Compressing after training methods focuses on the compressing work of already trained network. For example, [15] use low-rank decomposition of the weight matrix to reduce the number of parameters. Recently, [16] introduce a three-stage pipeline including pruning, trained quantization and Huffman coding, to reduce the storage requirement of networks without affecting their accuracy. Most relevantly, [17] introduce DCT and low-cost hash function to randomly group frequency parameters into hash buckets. Due to the lack of analysis of sparse property, compression results of this approach could be improved.

3. The proposed method

In this section, we describe CE-Text by lightweight CNN structure design, hierarchical attention scheme and embedded version design with deep compression.

3.1. Lightweight CNN structure design

Network design of the proposed method is presented in Fig. 2. Specifically, CE-Text would predict locations of N bounding boxes (representing different text components) inside each grid and the corresponding confidence about text existence c. Specifically, we define locations of bounding boxes as $B = \{x, y, w, h\}$, where x and y represent center positions of each box relative to grid boundary, and w and h refer to the predicted width and height of bounding box relative to the whole input image. Moreover, text existence confidence c is defined as

$$c = P(T) * \frac{area(B) \cap area(G)}{area(B) \cup area(G)} \quad (1)$$

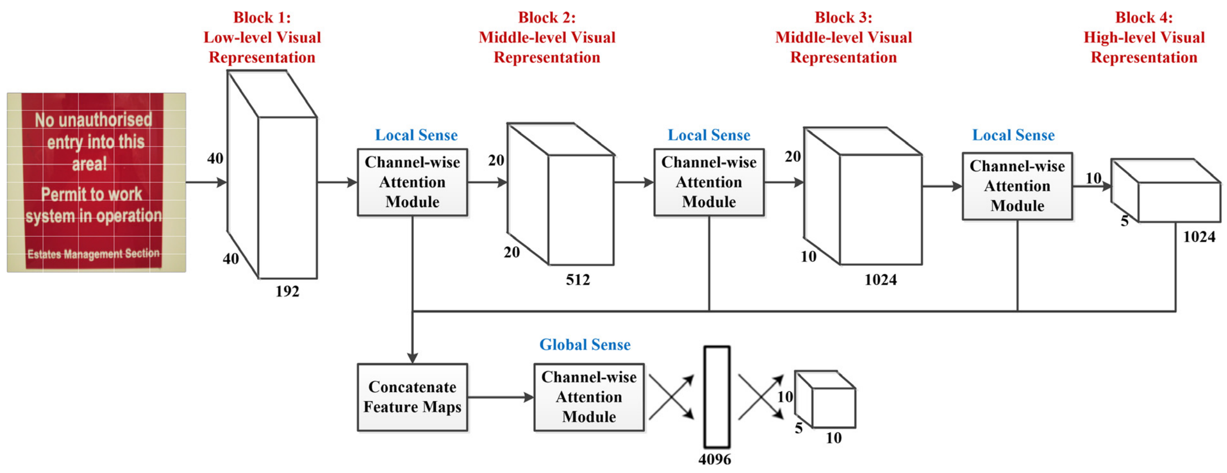

Fig. 2. Network architecture of the proposed CE-Text.

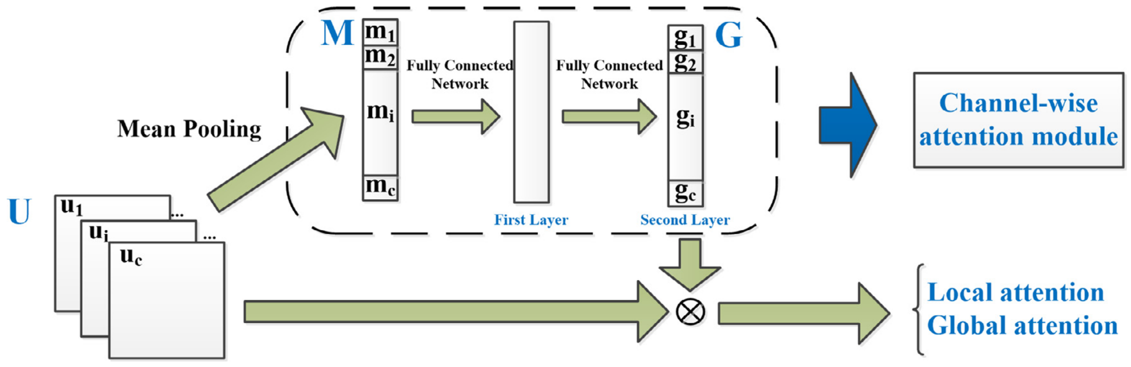

Fig. 3. Structure of the proposed channel-wise attention module.

where function *area*() help calculate areas of predicted bounding box B or ground-truth bounding box G. With such regression strategy of dividing input images into grids, regression results about text components can be encoded as a $W * H * (N * 5)$ tensor, where we set $W = 10$, $H = 5$ and $N = 2$ by experiments and 5 refers to the number of bounding box related elements, i.e., $\{x, y, w, h, c\}$.

To accurately perform regression task for locations and existences of text components, we build the proposed lightweight CNN structure on the basis of YOLOv2, which could perform task of accurate and fast object detection with low computation resource. Moreover, we try to involve characteristics of text components for higher accuracy in text detection.We show construction details of the proposed lightweight CNN structure in Table 1, which consists of 24 convolutional layers and 2 fully connected layers. Following the block splitting rule represented in VGG16 network, we divide convolutional and pooling layers into four blocks for the purpose of constructing hierarchical attention scheme.

Specifically, we use a reduction layer with 1*1 kernel size to integrate information of different channels followed by a convolutional layer with 3*3 kernel size to extract feature, rather than adopting inception modules used by GoogLeNet. The reason for such design lies in the fact that it could reduce computation cost and keep high capability to extract variety of features during training. It's noted that we adopt two pooling layer with 2*4 and 1*2 kernel size at the end of Block2 and Block3, which originates from the fact that text components generally have a large value in ratio between width and height. Therefore, such pooling kernels could help expand receptive field of the proposed CNN network in horizontal direction, which promotes for accurate detection of relative long text components. We utilize LeakReLU as activation function,

Table 1
Construction details for the network of CE-Text, where values of Type, such as Block 1, 2 and so on, can be found correspondences in Fig. 3.

Name	Type	Kernel No.	Kernel Size	Stride	Output Size
Block1	Conv	64	7*7	2	160*160*64
	Pooling		2*2	2	80*80*64
	Conv	192	3*3	1	80*80*192
	Pooling		2*2	2	40*40*192
Block2	Conv	128	1*1	1	40*40*128
	Conv	256	3*3	1	40*40*256
	Conv	256	1*1	1	40*40*256
	Conv	512	3*3	1	40*40*512
	Pooling		2*4	2,4	20*20*512
Block3	Conv	256	1*1	1	20*20*256
	Conv	512	3*3	1	20*20*512
	Conv	256	1*1	1	20*20*256
	Conv	512	3*3	1	20*20*512
	Conv	256	1*1	1	20*20*256
	Conv	512	3*3	1	20*20*512
	Conv	256	1*1	1	20*20*256
	Conv	512	3*3	1	20*20*512
	Conv	512	1*1	1	20*20*512
	Conv	1024	3*3	1	20*20*1024
	Pooling		1*2	1,2	20*10*1024
Block4	Conv	512	1*1	1	20*10*512
	Conv	1024	3*3	1	20*10*1024
	Conv	512	1*1	1	20*10*512
	Conv	1024	3*3	1	20*10*1024
	Conv	1024	3*3	1	20*10*1024
	Conv	1024	3*3	2	10*5*1024
	Conv	1024	3*3	1	10*5*1024
	Conv	1024	3*3	1	10*5*1024
FC-layer	FC				4096*1*1
	FC				10*5*10

which helps prevent information of output to be eliminated, efficiently updating parameters of neurons.

3.2. Hierarchical attention scheme

In this subsection, we introduce a channel-wise attention module as a lightweight gating mechanism, and it will be further utilized in a local and global manner to construct the hierarchal attention model.

We show the structure of proposed channel-wise attention module in Fig. 3. Specifically, let's suppose an input feature map $U = [u_i; i = 1, \dots, c]$, where u_i represents the ith channel of U and c is the total number of feature channels. Since each of the learned convolutional filters operate with a local receptive field to compute features and couldn't exploit contextual information outside of this region, we propose to utilize mean pooling operator to collect the contextual information into a channel descriptor, which could be represented as

$$m_i = \frac{1}{W \times H} \sum_{j=1}^{W} \sum_{k=1}^{H} u_i(j,k), \tag{2}$$

where W and H are width and height of u_i. M consists of m_i and is defined as the channel descriptor.

As shown in Fig. 3, the proposed channel-wise attention module is composed of two fully-connected layers and two corresponding nonlinear activation functions:

$$g_i = Nor(sig(W_2 \eta(W_1 m_i + b_1) + b_2)), \tag{3}$$

where function $sig()$, $Nor()$ and $\eta()$ refer to sigmoid, normalization and ReLU functions respectively, W_1 and W_2 are the learnable parameter matric, and b_1 and b_2 are the bias vectors. The reason to adopt such structure for module constructing lies in two facts, *i.e.*, firstly the designed structure must be capable of learning a highly nonlinear interaction between channels, and secondly, it must allow multiple channels to be emphasised opposed to one-hot activation. The resulting weight vector $\{G = g_i; i = 1, \dots, c\}$ thus leads to the attention on informative channel features, where Fig. 3 explains how the channel-wise attention module works with

$$\tilde{u_i} = u_i \otimes g_i \tag{4}$$

where $\otimes$ represents the element-wise multiplication.

To further involve distinguish characteristics of text, we construct a hierarchical attention scheme, which encodes text context-aware information into feature map by building channel-wise attention modules in both local and global sense. The reason to design hierarchical attention scheme lies in the fact that low-level and middle-level visual features could be more informative than high-level visual features for some cases of ambiguous text detection. Take one of typical low-level features, i.e., texture, as an example, similarity of texture between training and testing text components could be the most dominated and informative feature channel to accurately locate blur text components. However, the decay effect of gradients and semantic abstraction of higher layers in neural network may ignore low-level or middle-level visual features to a certain extent. We thus construct a hierarchical attention scheme to emphasize low-level and middle-level visual features to output text context-aware feature map for accurate text detection.

Specifically, we build local channel-wise attention modules at the end of blocks, which could utilize the channel-wise property of CNN features to automatically focuses on discriminative and representative features for text detection in a local sense. Local channel-wise attention weights are thus computed based on the lth channel feature $U_{k,l}$:

$$\tilde{U_{k,l}} = U_{k,l} \otimes \Phi(U_{k,l}) \tag{5}$$

where k is the index of block, Φ represents channel-wise attention module, and $\otimes$ is element-wise multiplication.

After representing local attention, we first concatenate features $\tilde{U_{k,l}}$ output by local attention modules and then utilize a channel-wise attention module to weight multi-layer CNN features, which could be written as :

$$\begin{cases} \tilde{U_k} = [\tilde{U_{k,l}}], l = 1, \dots, c \\ \hat{U_k} = \tilde{U_k} \otimes \Phi(\tilde{U_k}) \end{cases} \tag{6}$$

where function $[\cdot]$ represents operation of transforming and concatenating different size of matric into a single vector. Essentially, such global channel-wise attention module utilizes the multi-layer property of CNN features, in order to emphasize the representative ability of low and middle layers of visual cues. Based on text context-aware feature map, we finally apply average pooling operation and fully-connected layers to obtain predictions on text location and existence.

In inference process, non-maximum suppression is adopted to locate bounding box with the highest text existence confidence, thus eliminating other redundant bounding boxes. Widely used in edge detection, face detection and so on, non-maximum suppression algorithm could solve the problem of a large number of candidates centered at the same target position.

3.3. Embedded version design with deep compression

In this subsection, we aim to design a novel frequency-based CNN compression method, which could be applied on CE-Text to build a light scale version for deployment on embedded systems. The workflow of the proposed deep compression method is shown in Fig. 4, where (a) refers to the input parameter matrix, (b) use DCT to transform input matrix into frequency domain, (c) prune frequency matrix and save it for inference.

CE-Text network are composed of convolutional layers (COV), batch normalization layers (BN) and max-pooling layers (MP). It's noted that parameters in COVs and BNs require training. Therefore, the proposed compressing method constructs parameter matrix based on parameters of COVs and BNs. For each COV, we thus sperate them by blocks determined as 15*15 by experiments. Since each BN has three parameters, there are only 3*22=66 parameters in total for all the BNs. We thus use one 15*15 matrix to store the parameters of BNs. There would be blanks when transforming parameters of filters to blocks. We use the mean value of blocks to fill up these blanks, since the mean value won't change the frequency distribution. After separating and filling blanks, we could get a parameter matrix for each COV and BN by concatenating blocks. The parameter matrix is represented as N_l, where l represents the index of layers varying from 0 to 22. Note that index 0 represents the parameter matrix constructed based on parameters of BNs.

In our compressing method, DCT also suitable to compress extracted parameter matrix, since the weights in parameter matrices are typically smooth and low-frequency caused by the property of spatial locality of image pixels. Given an input matrix N, the corresponding frequency matrix M after DCT transforming could be written as follows:

$$M = ANA^T, where\ A(i,j) = c(i)\cos[\frac{(j+0.5)\pi}{d} i] \tag{7}$$

where d is defined as the length of input matrix N, i and j are the row and column index respectively, $c(i) = \sqrt{\frac{1}{d}}$ when $i = 0$ and $c(i) = \sqrt{\frac{2}{d}}$ when $i \neq 0$.

We prune frequency matrices to help save computation and storage cost. Essentially, network pruning has been widely studied to compress CNN models. The proposed method build on top of these approaches. Fig. 4 (a) and (b) show examples of frequency

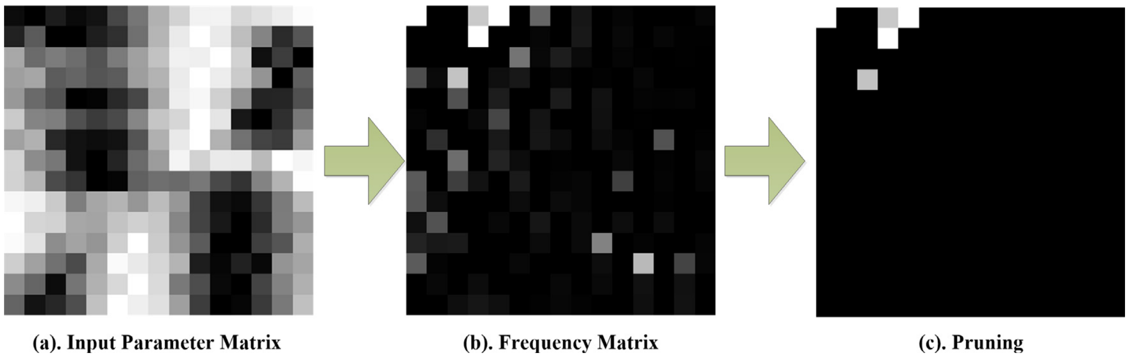

Fig. 4. Workflow of the proposed frequency-based compression method.

matrix in spatial and frequency domain, respectively. In the frequency domain, the upper left part with small indices (i, j), known as low-frequency components, have larger magnitude values than other parts named as high-frequency components. Based on this observation, we could conclude the energy of frequency matrix is dominated by low-frequency part. In other words, the upper left frequency values are more important than other values in constructing filters of CE-Text. To decrease storage and computation cost and maintain detection results for CE-Text, we should prune the high-frequency part and retain the low-frequency part. After removing high frequency components with a threshold α to define ratio of punning areas, we could get pruned and sparse result as shown in Fig. 4 (c).

4. Experimental results

4.1. Implementation details

Experiments are all carried out on a server, which is configured with Intel Xeon E5-2630v4 CPU (10cores and each with 2.2GHz), 64G memory and 1 piece of NVIDIA Titan X card. Meanwhile, we perform all embedded system related experiments on an ARM Cortex-A9 (4 cores @ 1.60GHz) Embedded system with 2G RAM. Training for CE-Text is performed by an Adam optimizer with adaptive learning rate, which is settled with an initial learning rate 0.001 and batch size 12. Compared to batch gradient descent method, Adam optimizer has more parameter setting with quantity of hyperparameters. However, it could be trained faster with significant convergence speed, which fits with the goal of the proposed method.

4.2. Results and analysis

The quantitative results of the proposed technique and the existing techniques are reported for different datasets, namely ICDAR 2011, ICDAR 2013 and SVT, inTables 2–4, respectively. The ICDAR 2013 dataset is similar to the ICDAR 2011 dataset and could be regarded as an extended version of ICDAR 2011. It is noted from the above standard datasets that characters suffer from low resolution, low contrast, multi-scripts and complex backgrounds. Also, the characters of natural scene data suffer from large font size variations. Especially, the characters of SVT suffer from severe complex backgrounds containing greenery, buildings, sky, etc.

In Tables 2–4, higher performance of CE-Text with attention, represented as CE-Text(wA), than without attention represented as CE-Text(woA) proves the efficiency and robustness of the designed attention module. Moreover, highly visual resemblance of text and non-text components could utilize low-level features, such as edge, texture and so on, to assist accurate detection by the proposed hierarchical structure. It's also noted that the hierarchical attention module only slightly increase the computation burden to exchange for much higher performance in both precision and recall. Above all, the proposed hierarchial attention model boosts the detection performance, which proves the effectiveness of CE-Text(wA).

We notice that methods generally achieve almost the same performance on ICDAR 2011 and ICDAR 2013, since both datasets are similar in contents and difficulties. It's noted that the large increases of computation time in [19], [20] and [21] lie in the fact

Table 2
Performance comparison with comparative text localization methods running on embedded systems for ICDAR 2011 dataset.

Methods	P	R	F	T(s)	FPS
[1]	0.86	0.74	0.80	0.110	9.09
[18]	0.81	0.61	0.70	0.137	7.30
[19]	0.76	0.65	0.70	0.148	6.76
[20]	0.75	0.63	0.68	0.447	2.24
[21]	0.81	0.66	0.73	3.812	0.26
[22]	0.86	0.76	0.81	1.502	0.67
[23]	**0.87**	**0.79**	**0.83**	2.207	0.45
CE-Text(woA)	0.73	0.72	0.73	**0.063**	**15.87**
CE-Text(wA)	0.76	0.75	0.76	0.065	15.38

Table 3
Performance comparison with comparative text localization methods running on PC platform for ICDAR 2013 dataset.

Methods	P	R	F	T(s)	FPS
[24]	0.48	0.47	0.47	0.136	7.35
[1]	0.86	0.74	0.80	0.113	8.85
[18]	0.80	0.60	0.68	0.132	7.58
[19]	0.74	0.65	0.69	0.169	5.92
[20]	0.76	0.62	0.68	0.434	2.30
[21]	0.81	0.66	0.73	3.705	0.27
[22]	0.85	0.76	0.80	1.489	0.67
[23]	**0.88**	**0.78**	**0.83**	2.147	0.47
CE-Text(woA)	0.73	0.71	0.72	**0.064**	**15.63**
CE-Text(wA)	0.76	0.74	0.75	0.065	15.38

Table 4
Performance comparison with comparative text localization methods on PC platform for SVT dataset.

Methods	P	R	F	T(s)	FPS
[24]	0.73	0.60	0.66	0.121	8.26
[1]	0.82	0.72	0.77	0.127	7.87
[18]	0.76	0.57	0.65	0.141	7.09
[19]	0.74	0.65	0.69	0.383	2.61
[20]	0.74	0.60	0.66	0.542	1.85
[21]	0.78	0.66	0.72	4.570	0.22
[22]	0.81	0.74	0.77	1.692	0.59
[23]	**0.84**	**0.76**	**0.80**	2.151	0.46
CE-Text(woA)	0.71	0.70	0.70	**0.067**	**14.93**
CE-Text(wA)	0.73	0.73	0.73	0.069	14.49

that they are MSER or ER based algorithms to generate text candidates. Complex background streetview images in SVT thus leads to a large amount of candidates to classify the existence of text, which increases the computation burden. CE-Text(wA) has almost constant performance on three different datasets to prove its robustness to deal with different types of inputting images.

It can be observed from all tables that Zhang et al. [23] is the best at both precision and recall compared to the other techniques. The same reason for the second best results achieved by Tian et al. [22], which utilizes min-cost cut to help accurately locate texts. However, the running time for both methods is much higher than regression-based methods, such as TextBox(S), CE-Text etc. The low fps speed of Zhang et al. [23] is second smallest among all methods, only slightly better than [21]. Such methods are not suitable for fast purpose due to high computation time[21]. apply multiple processing stages to perform text detection task, which is difficult in optimization and leads to the largest running time. Compared with these three methods, CE-Text(wA) achieve the highest fps values and acceptable detections results, which makes it suitable to apply for fast applications.

Among comparative methods, Wang et al. [19] and Li et al. [20] utilize manually-design features to locate text, which achieve shorter running time than most of deep neural network based methods. However, they achieve lower performance in precision and recall, due to the lack of powerful and discriminative features for detection. Compared with the proposed method, CE-Text(wA) outperforms these two methods not only in accuracy, but also in running time, which owes to the high performance of Cuda architecture and Titan hardware adopted by CE-Text(wA), complex and individual steps of Wang et al. [19] and Li et al. [20] with high difficulty to optimize, and implementations with Matlab making [20] difficult to achieve high runtime performance.

We could notice that Liu et al. [18] and Liao et al. [1] share the idea of directly utilizing regression for text location with CE-Text and have advantages of low computation cost as well. Specially, we can notice [18] and [1] have almost the same computation time, since Textbox(S) is designed on the basis of SSD. However, SSD aims to solve problems of common object detection, which achieves lower performance than Textbox(S) and CE-Text(wA), due to ignoring special characteristics of texts. Textbox(s) achieves higher performance than CE-Text(wA) in precision. Recall that Textbox(s) involves the recognition of words in dictionary to help correct detections, we think it's fair for CE-Text(wA) to get a lower performance. However, Textbox(s) suffers from low recall performance and almost two times larger running time than CE-Text(wA). By reducing a large amount of parameters, CE-Text(wA) is lighter in size and much faster in speed than Textbox(s), which makes it quite suitable to be applied in tasks with high requirements for fast performance. Above all, we could draw a conclusion that under restrictions or scenarios of fast computing, CE-Text(wA) is appropriate to be adopted for task of text detection in scene image, since it keeps a reasonable balance between accuracy and computation cost.

Since the goal of the proposed method is to process each input image with real-time feedback, we design the proposed network with simple but effective structure. Under such design principle, we achieve worse performance on SVT dataset on PC platform, which guarantees sufficient computing resources for text detection task. However, the proposed network achieves successful implementation in embedded system. Meanwhile, most of the comparative and deep learning based methods fail to be successfully implemented due to limited computation resource, especially enough memory. All these facts can be proved by the results in Table 5. Examples of qualitative results of the proposed technique for different datasets are shown in Fig. 5, where it can be seen that texts are detected from video, scene images and street view images successfully. This shows that the proposed technique helps in achieving good text detection results.

Table 5
Performance comparison with comparative text localization methods running on embedded systems for ICDAR 2013 dataset.

Methods	P	R	F	T(s)	FPS	Size
[1]	*fail*	*fail*	*fail*	*fail*	*fail*	*fail*
[18]	*fail*	*fail*	*fail*	*fail*	*fail*	*fail*
[19]	**0.74**	0.65	0.69	**0.427**	**2.34**	**43MB**
[20]	*fail*	*fail*	*fail*	*fail*	*fail*	*fail*
[21]	*fail*	*fail*	*fail*	*fail*	*fail*	*fail*
[23]	*fail*	*fail*	*fail*	*fail*	*fail*	*fail*
CE-Text($\alpha = 0.3$)	0.75	**0.73**	**0.74**	0.531	1.88	147MB
CE-Text($\alpha = 0.4$)	0.74	**0.73**	0.73	0.529	1.89	127MB
CE-Text($\alpha = 0.5$)	0.74	0.72	0.73	0.524	1.91	106MB
CE-Text($\alpha = 0.6$)	0.68	0.64	0.66	0.521	1.92	86MB

4.3. Results and analysis with embedded version

Since ICDAR 2011 and ICDAR 2013 are similar in content and SVT contains only street view images, we choose ICDAR 2013 to perform embedded system experiments, which could reflect the performance of text detection on focused scene and street view images. Table. 5 gives the detailed statics of CE-Text and other comparative methods on ICDAR 2013 dataset, where measurement *Size* represents deployment storage size, α refers to the prune ratio and we choose the compression version of CE-Text(wA) as the compared CE-Text. It's noted that we come across many failure cases when transforming comparative methods from Intel version to Arm version. However, fail reasons are different from case to case, where Textbox(S) [1] fails due to its publish code is short of several key parts for running on Arm-based system, implementations of SSD [1], Wu et al. [21] and FCN [23] come across the problem of out of memory due to its high request for memory to store parameters, and Li et al. [20] fails since its code version is matlab that we can't transform it to run on Arm-based system successfully. We must point out there are CPU version of SSD to run. To keep version same for fairness of comparative studies, we use the original version of SSD for experiments in Table. 5.

From Table. 5, we can see Wang et al. [19] keeps almost constant performance on both GPU and embedded systems, where the increase in running time can be explained as the difference of CPU clock speed of these systems. The reason of Wang et al. [19] to retain constance lies in the fact that it simply utilizes one distinguish feature for detection and its procedures are designed with low computation cost for fast purpose. Compared with CE-Text, it still suffers from slightly lower performance and the fact, that upper bounder of Wang et al. [19] can't be easily improved by utilizing GPUs or high performance CPUs due to its complex and individual steps with high difficulty to optimize. It's clear from Table. 5 that our method suffers from the transformation from GPU to CPU architecture as well, which results in much higher computing time. However, the proposed method get almost the same running performance with Wang et al. [19] due to the compression of CE-Text to make it smaller in both storage and memory size.

It's noted that we could get different performance, especially precision, recall, f-measure and storage size, with different settings of prune ratio α. Generally speaking, high value of α leads to smaller storage size and lower performance of CE-Text. We thus need keep a balance between storage size and runtime performance. Specifically, we could find that CE-Text gets stable performance when α increases from 0.3 to 0.5. The performance of CE-Text drops greatly when α increases from 0.5 to 0.6. The inconsistent performance of CE-Text could be explained by the fact that performance would drop greatly if the proposed compression method prunes relatively low-frequency part of parameter matrix.

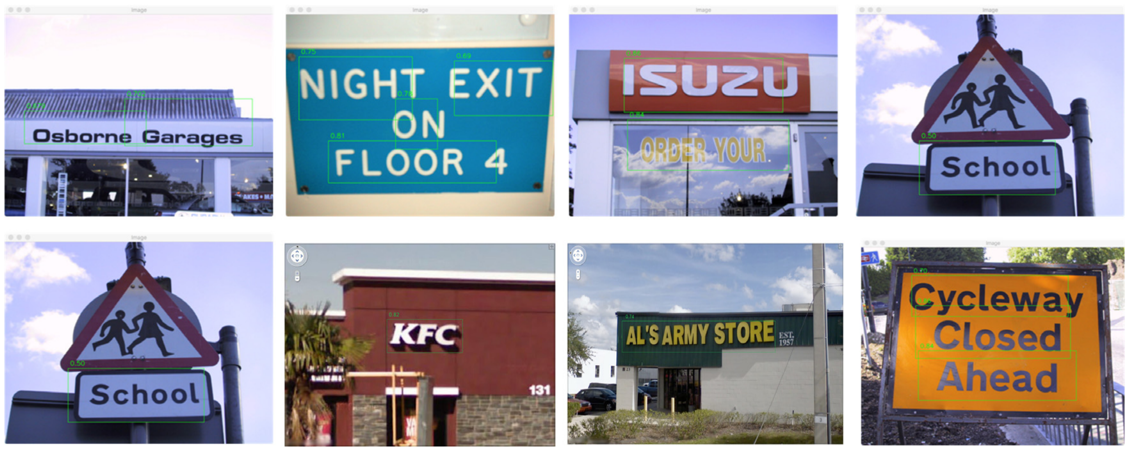

Fig. 5. Samples of text detection results achieved by CE-Text on ICDAR 2011, 2013 and SVT datasets.

Therefore, we settle $\alpha = 0.5$ with 106MB deployment storage size for embedded version, which can perform text detection task on embedded systems with high accuracy and running speed.

5. Conclusion and future work

In this work, we propose a lightweight and context-aware deep convolutional neural network (CNN) for text detection. The proposed method proposes a hierarchical text attention scheme, which captures context information by constructing multi-level channel attention modules. To fit with low computation resource of embedded systems, we further transform CE-Text into a lighter version with a frequency based deep CNN compression method. Experimental results on both workstations and embedded systems demonstrate the effectiveness and robustness of CE-Text for text localization task.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by National Key R&D Program of China under Grant 2021YFB3900601, the fundamental research funds for the central universities (32519113008,32519113006, 31512111310,B220202074) and by the open project from the State Key Laboratory for Novel Software Technology, Nanjing University, under Grant No. KFKT2019B17.

References

[1] M. Liao, B. Shi, X. Bai, X. Wang, W. Liu, Textboxes: a fast text detector with a single deep neural network, in: Proceedings of AAAI, 2017, pp. 4161–4167.
[2] C. Chen, L. Liu, S. Wan, X. Hui, Q. Pei, Data dissemination for industry 4.0 applications in internet of vehicles based on short-term traffic prediction, ACM Transact. Internet Technol. (TOIT) 22 (1) (2021) 1–18.
[3] H. Wang, D. Zhang, S. Ding, Z. Gao, J. Feng, S. Wan, Rib segmentation algorithm for x-ray image based on unpaired sample augmentation and multi-scale network, Neur. Comput. Applic. (2021) 1–15.
[4] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, J. Liang, EAST: an efficient and accurate scene text detector, in: Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2642–2651.
[5] B. Shi, X. Bai, S.J. Belongie, Detecting oriented text in natural images by linking segments, in: Proceedings of CVPR, 2017, pp. 3482–3490.
[6] C. Yao, X. Bai, N. Sang, X. Zhou, S. Zhou, Z. Cao, Scene text detection via holistic, multi-channel prediction, CoRR abs/1606.09002 (2016).
[7] Y. Wu, P. Natarajan, Self-organized text detection with minimal post-processing via border learning, in: Proceedings of 2017 IEEE International Conference on Computer Vision, 2017, pp. 5010–5019.
[8] L. Wu, C. Quan, C. Li, Q. Wang, B. Zheng, X. Luo, A context-aware user-item representation learning for item recommendation, ACM Transact. Inform. Syst. (TOIS) 37 (2) (2019) 1–29.
[9] S. Wan, S. Ding, C. Chen, Edge computing enabled video segmentation for real-time traffic monitoring in internet of vehicles, Pattern Recognit. 121 (2022) 108146.
[10] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, C. Yao, Textsnake: A flexible representation for detecting text of arbitrary shapes, in: Proceedings of 2018 European Conference on Computer Vision, 2018, pp. 19–35.
[11] P. Lyu, C. Yao, W. Wu, S. Yan, X. Bai, Multi-oriented scene text detection via corner localization and region segmentation, in: Proceedings of 2018 IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7553–7563.
[12] S. Ding, S. Qu, Y. Xi, S. Wan, Stimulus-driven and concept-driven analysis for image caption generation, Neurocomputing 398 (2020) 520–530.
[13] A. Zareian, S. Karaman, S. Chang, Bridging knowledge graphs to generate scene graphs, in: Proceedings of European Conference on Computer Vision, volume 12368, 2020, pp. 606–623.
[14] M. Courbariaux, Y. Bengio, Binarynet: training deep neural networks with weights and activations constrained to +1 or -1, CoRR abs/1602.02830 (2016).
[15] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, N. de Freitas, Predicting parameters in deep learning, in: Proceedings of 2013 Neural Information Processing Systems Conference, 2013, pp. 2148–2156.
[16] S. Han, H. Mao, W.J. Dally, Deep compression: compressing deep neural network with pruning, trained quantization and huffman coding, CoRR abs/1510.00149 (2015).
[17] W. Chen, J.T. Wilson, S. Tyree, K.Q. Weinberger, Y. Chen, Compressing convolutional neural networks in the frequency domain, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1475–1484.
[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S.E. Reed, C. Fu, A.C. Berg, SSD: single shot multibox detector, in: Proceedings of 14th European Conference on Computer Vision, 2016, pp. 21–37.
[19] W. Wang, Y. Wu, S. Palaiahnakote, T. Lu, J. Liu, Cloud of line distribution for arbitrary text detection in scene/video/license plate images, in: Proceedings of Pacific-Rim Conference on Multimedia, 2017, pp. 433–443.
[20] Y. Li, W. Jia, C. Shen, A. van den Hengel, Characterness: an indicator of text in the wild, IEEE Trans. Image Process. 23 (4) (2014) 1666–1677.
[21] Y. Wu, W. Wang, S. Palaiahnakote, T. Lu, A robust symmetry-based method for scene/video text detection through neural network, in: Proceedings of International Conference on Document Analysis and Recognition, 2017, pp. 1249–1254.
[22] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, C.L. Tan, Text flow: A unified text detection system in natural scene images, in: Proceedings of 2015 IEEE International Conference on Computer Vision, 2015, pp. 4651–4659.
[23] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, X. Bai, Multi-oriented text detection with fully convolutional networks, in: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4159–4167.
[24] B. Epshtein, E. Ofek, Y. Wexler, Detecting text in natural scenes with stroke width transform, in: Proceedings of 2010 IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2963–2970.