Text Recognition Using Local Correlation

Yujia Li[12]
liyujia@iie.ac.cn
Hongchao Gao[12]
gaohongchao@iie.ac.cn
Xi Wang[1]
wangxi1@iie.ac.cn
Jizhong Han[1]
hanjizhong@iie.ac.cn
Ruixuan Li[13]
ruixuanli@iie.ac.cn

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China

Abstract

In this paper, we propose an improved scene text recognition method by considering the local correlation of the character regions. Fractal theory indicates that most images have self-similarity properties including scene text images. The recent methods always extract text representations of scene image through Convolution Neural Networks (CNN) which use fixed kernels. The self-similarity of the image is not fully considered. In our paper, we propose a Local Correlation (LC) representation mechanism which capture richer text representations by considering the local correlation of the character regions. This mechanism not only brings significant improvement of recognition results but also can be easily embedded in other recognition architectures. After we embed this mechanism in scene text recognition architecture, the experiment on several benchmark datasets (IIIT-5K, SVT, CUTE80, SVT-Perspective, ICDAR) shows that the proposed architecture can obtain richer representations of the scene images and strikes a good efficiency/accuracy trade-off.

1 Introduction

Text is ubiquitous in our daily life. It can be found on documents, road signs, and other objects like cars or telephones. The goal of scene text recognition is to map an input scene text image into a set of character sequences. Our work aims to recognize a text string from a cropped word image. The existence of complex environments, such as uneven lighting and position changes, blurring, occlusion and so on, makes recognition of the scene text images to be a challenging topic. The first step of text image recognition is usually extracting the context information from images. Then, transform this information to text. Most previous approaches often segment a cropped image into many character patches, then followed by an isolated character classifier and post-processing for recognition [29, 30]. These approaches mainly focus on developing powerful character classifiers to improve the final recognition performance, some of which are incorporated with a language model, leading to the state-of-the-art performance [4, 16, 29, 30, 45]. They also adopt deep neural networks for representations learning, but the recognition is still confined to character-level classification [13]. Thus, their performance is severely harmed by the unsuitable single character segmentation especially when recognizing irregular images. Importantly, recognizing each character independently discards meaningful context information of the words, significantly reducing its reliability and robustness. With the success of sequence modeling on audio synthesis, word-level language modeling and machine translation [38, 44], a lot of works have been proposed for text recognition to deal with the above problems [6, 7, 34, 35, 39]. Those methods try to make the text recognition as a sequence modeling task, which does not need to separate the single characters. These kinds of methods usually contain two stages. As shown in Figure 1.

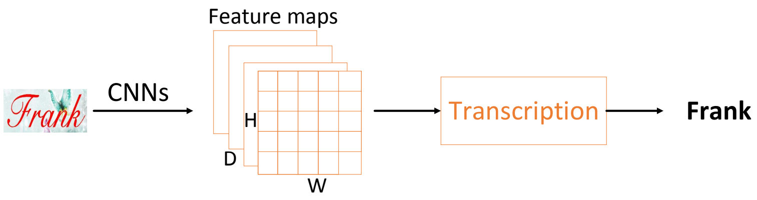

Figure 1: The network architecture.

The first stage is to extract the feature sequences of input text image using the CNNs. For example, in [35], Shi et al. extracted the feature maps from a text image by VGG-net[37] and split those into a sequence of feature vectors. There are some other methods to try to improve the sequential feature representations on irregular text. For example, Bartz et al. [3] trained a network to obtain an affine transformation [17] to align the direction of the text.

The second stage is to transform the input sequence of feature vectors to an output sequence of characters. Recurrent Neural Network (RNN) is used to generate the target characters based on the glimpse vectors and the history of previous output characters [35, 39]. Most popular methods use the encoder-decoder architecture [1, 10, 38], which is introduced from the language translation. For example, the recognition models designed in [6, 34] used the recurrent neural architecture proposed by [1] and Gao et al. [9] designed the recognition model based on the convolutional sequence architecture proposed in [10].

Most existing scene text recognition networks focus on improving the second stage which is based on the attention mechanism. For the first stage, most methods often use the traditional CNNs to extract the features of text images. We notice that text images are self-similarity indicated by the fractal theory [26]. However, this self-similarity property is not fully represented by the CNNs which use fixed kernels. In our paper, we focus on extracting richer representations of the word images. A Local Correlation mechanism is proposed to make full use of the self-similarity of the text image. Contributions of this paper are described as follows:

- We propose LC representation mechanism that can make use of local correlation information between adjacent vectors to get better feature representations and the proposed mechanism can be easily embedded in the other encoder-decoder architectures;
- We develop a novel scene recognition architecture (LC-Net) based on this mechanism,

which strikes a good efficiency/accuracy trade-off on several benchmark datasets;

- Our proposed architecture can recognize image with arbitrary shape and proportion.

2 Related Work

Scene text recognition. This task converts scene text images into strings. With the success of deep neural networks in computer vision tasks, many methods adopt deep neural networks for text recognition. Basically, there are two types of approaches: bottom-up and top-down [7, 8]. Bottom-up methods extract low-level features from individual character region, then detect and recognize these characters one by one, finally, integrate them basing on language model [45]. Top-down methods predict the entire text from the original image without detecting the characters [27]. Recent researches solve this problem as a sequence recognition problem. For example, the input sequences are regarded as a group of knowledge representations of image patches to be recognized [7, 8, 35, 36]. Our method falls into the category of sequence recognition. We use the LC representation mechanism as the encoder to map image patches into a sequence of continuous representations, then use decoder layers to generate the corresponding text.

Local correlation representations. Fractal theory indicates that most images have self-similarity properties and adjacent pixels have the same semantic information [26]. For example, those pixels can collectively represent a character in scene text images. Rich knowledge representations can be obtained from neighbor pixels. These representations can be used for interest point detectors [21], image denoising [5] and video classification [42]. For example, [5] proposed digital image denoising method based on non-local averaging of all pixels in the image, which was a classical filtering algorithm that computed a weighted mean of all pixels in an image. Furthermore, Wang *et al.* [42] proposed video classification methods based on non-local operation and computed the response at a position as a weighted sum of the features at all positions in input feature maps. In this paper, based on attention mechanism and local correlation, we propose an LC representation mechanism to extract more meaningful representations of current pixels.

Attention mechanism. Recently, attention mechanism has become an integral part of scene text detection [14] and recognition [7, 9, 22], resulting in significant improvements in such tasks. Attention mechanism focuses on the most important segments of incoming features and adds a degree of model interpretability [25, 34, 38]. Lee et al. [6] used a soft-attention mechanism to allow the model to selectively exploit image features in a coordinated way. Liu et al. [22] proposed a model composed of word-level and char-level encoder. Cheng et al. [7] added localization supervision to the attention module to solve the attention drift problem in existing attention-based methods. Self-attention is an attention mechanism that associates different locations of a single sequence to calculate the representations of the sequence [25, 38]. A self-attention module computes the response of each position in a sequence by attending to all positions and taking their weighted average in an embedding space. [25] used the local attention mechanism in decoder layers to choose to focus only on a small subset of the source positions per target word. In this paper, we use self-attention operations in our LC representation mechanism to make full use of local correlation between adjacent pixels.

3 Local Correlation Network

In this paper, we present a scene text recognition architecture which is named LC-Net with three components, as shown in Figure 2, including the convolutional layers, Local Correlation representation mechanism (LC representation mechanism) which consists of three Local Correlation blocks (LC block) and a transcription layer. Given the image to be recognized, the network first extracts the feature maps with the convolution layers at the bottom, and splits it into a sequence of feature vectors from left to right. Then, LC representation mechanism encodes the extracted feature vector only using its adjacent vectors. Finally, a recurrent network is constructed to predict each feature vector encoded by the LC representation mechanism. Each feature vector prediction of the recurrent network is transformed into a tag sequence using the transcription layer at the top of LC-Net.

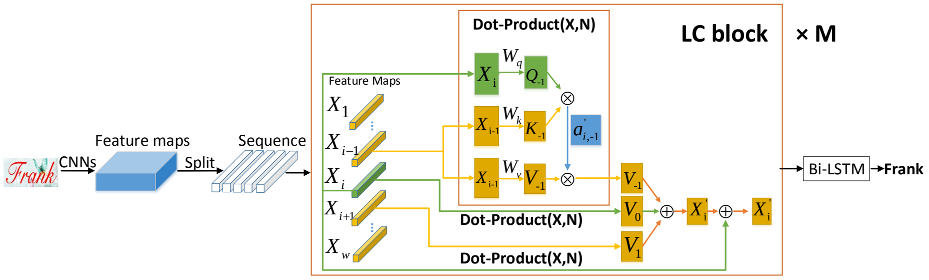

Figure 2: A local correlation network (LC-Net). We illustrate the figure with setting neighbor vectors to 3 (including itself). $\otimes$ denotes matrix multiplication, and $\oplus$ denotes element-wise sum. The softmax operation is performed on each row. M means the number of LC block.

3.1 Feature Sequence Extraction

In the LC-Net, feature sequence extraction layer is based on VGG-net[37] or ResNet(the configurations of layers the same as [36]). Before being entered into the network, the images need to be resized to 32×100. The feature maps generated by the last convolution layer is split into feature sequence by column from left to right and the width of each column in our settings is fixed to single pixel. So the item of the ith feature vector in feature sequence corresponds to the ith column in feature maps. Due to only using convolution and max-pooling operations on local regions, each vector represents a receptive field of the input image, and can be considered as the image descriptor for that region [35, 39]. As illustrated in Figure 2, several vectors represent a target character in the text image. Thus, the extracted feature sequence represents the whole image which is the input for the LC representation mechanism.

3.2 Local Correlation Representation Mechanism

As shown in Figure 2, continuous adjacent feature vectors correspond to the same character 'F', in other words, these vectors provide context information to each other. Our proposed LC representation mechanism which consist of three LC blocks can take advantage of the adjacent feature vector of current feature vector to extract rich knowledge representations of current feature vector in feature maps. We split feature maps into feature sequence X = $\{X_1, X_2, ...X_j, ...X_w\}$ which is the input for LC block, w is the width of last feature map. That is, the item of the ith feature vector X_i corresponds to the ith column in feature maps. Let $\{X_{i+j} | -k <= j <= k\}$ represents X_i neighbor vectors, and k represents restricted range of the number of adjacent vectors. For example, $k = 2$ represents that we will consider 4 adjacent vectors (2 feature vectors on the left side and 2 on the right side of the current one) while encoding the current vector. Given vector X_i and one of its neighbor vectors X_{i+j}, we use 3 learnable weight matrices(W_q,W_k,W_v) for local linear embedding to extract rich representations: $Q_i = W_q X_i$, $K_j = W_k X_{i+j}$, $V_j = W_v X_{i+j}$. Those can be implemented as $1 \times 1 \times 1$ convolution operations. And Q_i, K_j, V_j can be obtained through the backpropagation of the entire model training process. Besides, X_{i+j} is weighted twice using W_k and W_v, which can project the adjacent vector into different representation subspaces to obtain more feature representation.

The local correlation between vector X_i and its adjacent vector X_{i+j} is bigger, the weight assignment on X_{i+j} is greater. So, in linear projection space, we define a generic LC operation in deep neural networks as Eq.(1)(2).

$$X_i^{'} = \sum_{j=-k}^{k} (a_{i,j}^{'} V_j) \tag{1}$$

After element-wise sum and layer normalization, the output vector is obtained:

$$X_i^{'} = norm(X_i^{'} + X_i) \tag{2}$$

$a_{i,j}^{'}$ represents local correlation evaluation. In LC representation mechanism, we use pearson correlation coefficient to measure the degree of correlation between X_i and one of its adjacent vectors X_{i+j}, which can be seen in alignment model used in attention mechanism [38]. It can be proved that this evaluation is equal to the cosine similarity after subtracting the mean (centralized). Furthermore, the cosine similarity measure of two vectors is equivalent to the operator of the dot product of two vectors, as shown in Eq.(3). In practice, the vector dot product is implemented with highly optimized matrix multiplication code, so the calculation speed and space efficiency of this layer are very high.

$$a_{i,j} = \frac{Q_i \cdot K_j}{\delta_{Q_i} \delta_{K_j}} \tag{3}$$

δ_{Q_i}, δ_{K_j} are variance of vector. Finally, we use softmax function to transform the similarity between vectors to obtain the corresponding weight assignment:

$$a_{i,j}^{'} = \frac{a_{i,j}}{\sum_{j=-k}^{j=k} exp\,(a_{i,j})} \tag{4}$$

i corresponds to the ith feature vector X_i.

3.3 Transcription

We use bi-directional LSTM to generate predictions for each feature vector encoded by LC block and use Connectionist Temporal Classification (CTC) [11] to compute loss. Bi-directional LSTM can well capture the context information and backpropagation error difference to its input, namely the LC block, so we can train bi-directional LSTM layer and

LC block together in a unified network. Specifically, we use the negative log-likelihood of the highest sequence probability conditioned on every frame prediction which is defined in CTC as loss function, and combine the backpropagation algorithm to train the entire network including our proposed LC representation mechanism. With CTC, we only need the entire image and label sequence and do not need to label position of each single character, which avoids the requirements like a lot of labeling work, pre-separation of training data and post-output processing.

4 Experiment

We evaluate our system on seven scene text datasets. The different datasets we used and the specific introductions are given in Section 4.1. In Section 4.2, we give the implementation details of our experiment. In Section 4.3 we present a comparison of our methods with other methods on different datasets.

4.1 Datasets

Seven popular benchmarks for scene text recognition are used for performance evaluation, which are listed as follows.

IIIT5k [28] (IIIT 5K) is collected from the Internet, containing 3000 cropped word images. In addition, each image is associated with 50-word and 1K-word lexicon;

Street View Text [40] (SVT) contains 647 cropped word images which are collected from Google Street View. In addition, each image has a 50-word lexicon;

ICDAR 2003 [24] (IC03) contains 251 scene images which have bounding boxes of labeled text and 860 cropped images of the words;

ICDAR 2013 [18] (IC13) contains 1015 cropped text images. This dataset inherits most data in IC03;

ICDAR 2015 Text [19] (IC15) is generated by a pair of Google Glasses with blurry positioning and focusing. Besides, it has many irregular text, which increases the difficulty for recognition;

SVT-Perspective [31] (SVTP) The dataset is collected from lateral view of Google street. The dataset contains 639 cropped images and each image has been severely distorted;

CUTE80 [32] (CUTE) is mainly used for curved text, containing 80 high-resolution images taken in natural scenes. We crop the annotated words and get a test set of 288 images.

We use two training datasets including Synth90K [27] which contains 9 million images, and SynthText [12] which is the synthetic text dataset. For evaluation on the IC03, IC13 and SVT datasets, we discard all images that contain non-alphanumeric characters and images that have less than 3 characters as done in [3, 34, 35].

4.2 Implementation Details

Network: As shown in Figure 2, given the converted gray-scale images, we resize it to 32×100 and use VGG-net or ResNet to extract the feature maps of the image. The kernel size of the 3th and the 4th max pooling layers is changed from 2×2 to 1×2 to match English text recognition task. The size of feature maps is $26 \times 1 \times 512$. Then, we squeeze it to a feature map with scale 26×512 and split it to feature sequences that contain 26 vectors. The LC representation mechanism is used to encode those feature vectors, which is a stack

Number of LC block	IIIT5K	SVT	IC03	IC13	IC15	SVTP	CUTE
0	80	74.2	85.4	84.7	63.5	66.9	53.1
1	80.2	80.9	89.4	87.9	63.3	67.5	58.7
2	82.1	81.5	89.8	88.8	64.6	68.4	60.1
3	**83**	**83.5**	**91.1**	**88.9**	**68.1**	**69.1**	**64.9**

Table 1: Effects of using different number of LC block.

of 3 identical LC blocks, and the restricted range $k = 4$. The bi-directional LSTM layer is decoder layer and the final output is fully-connected linear projections with 36 units (26 letters, 10 digits).

Model training: The Synth90K and ST are used as training datasets. We use the Adam optimizer [20] with the learning rate $lr = 2^{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\varepsilon = 10^{-8}$. Our method is implemented under the Pytorch framework and trained on a computer with a Tesla M40 GPU. The batch size is set to 128 and training is stopped after 3 epochs over the training set. The inference speed is 11ms per image when the test batch size is 1 and this speed could be boosted with greater batch size.

4.3 Ablation Experiment on Local Correlation mechanism

A major contribution of our approach is proposing LC representation mechanism which is a stack of 3 identical LC blocks for getting better feature representations. In order to measure the contributions of our LC representation mechanism, we progressively increase the number of LC blocks and compare the results. For all the experiments in this subsection, we only use Synth90K as training dataset and use the same experimental settings, except for specified changes to the settings or component(s). Besides, we do not use lexicon on all benchmark datasets when we evaluate our LC representation mechanism on this experiment. As shown in Table 1.

Our method without LC block can be treated as Shi et al [35], As shown in Table 1. The performance is the worst when we do not use LC block. When we add 1 LC block to our method, the accuracy improves by an average of 2.9%. In particular, there is a 6.7% increase on SVT dataset and 5.6% increase on CUTE dataset. And if we use 2 LC blocks, the performance is 3.9% higher than that without LC block. When we use 3 LC blocks, the performance is improved by 2.9% on average compared with only using 1 LC block, and by 1.9% on average compared with using 2 LC blocks. It shows that the LC block we proposed can effectively improve the accuracy of text recognition. The main reason for this phenomenon is that LC block can make full use of local correlation between adjacent pixels and obtain better feature representations for text recognition.

4.4 Performance on General Recognition Datasets

In this section, we show the performance of our method and compare with other recent methods. The comparison results are shown in Table 2. We test our method under three different conditions. LC-VGG-A means using the VGG-net as backbone net and 90k as training data. In LC-VGG-B, the training data includes 90k and ST. And LC-RES use ResNet and 90k. The evaluation is taken on seven datasets. We can see that our method achieves the state-of-the-art performance on the 3 of 7. As the following, we divide the datasets into regular datasets and irregular datasets, to analyze our experimental results.

Methods	ConvNet, Data	IIIT5k			SVT		IC03			IC13	IC15	SVTP	CUTE
		50	1k	0	50	0	50	Full	0	0	0	0	0
Wang et al.[40]	-	-	-	-	57.0	-	76.0	62.0	-	-	-	-	-
Mishra et al.[29]	-	64.1	57.5	-	73.2	-	81.8	67.8	-	-	-	-	-
Wang et al.[41]	-	-	-	-	70.0	-	90.0	84.0	-	-	-	-	-
Bissacco et al.[1]	-	-	-	-	-	-	90.4	78.0	-	87.6	-	-	-
Yao et al.[45]	-	80.2	69.3	-	75.9	-	88.5	80.3	-	-	-	-	-
Jaderberg et al.[16]	-	-	-	-	86.1	-	96.2	91.5	-	-	-	-	-
Jaderberg et al.[17]	VGG, 90k	97.1	92.7	-	95.4	80.7	98.7	**98.6**	93.1	90.8	-	-	-
Jaderberg et al.[15]	VGG, 90k	95.5	89.6	-	93.2	71.7	97.8	97.0	89.6	81.8	-	-	-
Shi et al.[35]	VGG, 90k	97.6	94.4	78.2	96.4	80.8	98.7	97.6	89.4	86.7	-	-	-
*Shi et al[34]	VGG, 90k	96.2	93.8	81.9	95.5	81.9	98.3	96.2	90.1	88.6	-	71.8	59.2
Lee et al.[23]	VGG, 90k	96.8	94.4	78.4	96.3	80.7	97.9	97.0	88.7	90.0	-	-	-
Yang et al. [43]	VGG, Private	97.8	96.1	-	95.2	-	97.7	-	-	-	-	75.8	69.3
Wang et al. [39]	-, 90k	98.0	95.6	80.8	96.3	81.5	98.8	97.8	91.2	-	-	-	-
Cheng et al.[7]	ResNet, 90k+ST$^+$	99.3	97.5	87.4	97.1	85.9	99.2	97.3	94.2	93.3	70.6	-	-
Shi et al.[36]	VGG, 90k	98.1	95.7	81.7	97.6	85.5	98.7	97.3	92.2	88.6	67.6	73.2	63.9
Shi et al.[36]	ResNet, 90k	98.7	96.3	83.2	**99.2**	**87.6**	99.1	97.6	92.4	89.7	68.9	75.4	67.4
Cheng et al.[8]	-, 90k+ST$^+$	99.6	98.1	87.0	96.0	82.8	98.5	97.1	91.5	-	68.2	73.0	**76.8**
Liu et al.[22]	ResNet, 90k+ST$^+$	-	-	83.6	-	84.4	-	-	91.5	90.8	60.0	73.5	-
Fang et al.[33]	ResNet, 90k+ST	98.5	96.8	86.7	97.8	86.7	**99.3**	98.4	**94.8**	**93.5**	71.2	-	-
LC-VGG-A	VGG, 90k	98.6	97	83	96.5	83.5	98	96.2	91.1	88.9	68.1	69.1	64.9
LC-RES	ResNet, 90k	98.2	96.4	84.9	95.2	82.4	98.2	96.2	91.8	90.1	68.6	70.9	66.0
LC-VGG-B	VGG, 90k+ST	**99.6**	**98.2**	**89.9**	97.1	87.2	98.6	97	93.3	92.9	**74.5**	**76.4**	70.8

Table 2: The text recognition accuracy on general benchmarks. '50', '1k', 'Full' are lexicons. '0' means no lexicon. '*' means the conference version of this paper. '90k' and 'ST' are the Synth90k and the SynthText datasets, respectively. 'ST$^+$' means including character-level annotations. 'Private' means private training data.

Performance on regular datasets. The regular datasets include IIIT5K, SVT, IC03 and IC13. The results are shown in Table 2. On IIIT5K, our method achieves the best performance to the existing methods. With 50 word lexicon, we achieve the 99.6% recognition accuracy, which is the highest in all existing methods on seven datasets. In particular, without using word lexicon, we have a 2.5% improvement over the current best result [7] on IIIT5k dataset.

By using the same network bone and training data, our method LC-VGG-A and LC-RES achieve better performances than Shi et al.[36] with VGG-net and Shi et al. [36] with ResNet . For Shi et al.[36], there is a major factor leads to high performance: using an additional rectification network before recognition. However, rectifying images takes a lot of time and computing resources , so the inference speed of ours is 2 times faster than Shi et al. [36](as shown in Table 3). When using 90K+ST as training datasets, our method LC-VGG-B achieves state-of-the-art performance, and only falls short on a few results compared to [7]. But it's worth noting that [7] uses extra character-level annotations and uses additional Focusing Attention Network(FN) to draw back the drifted attention which will spend a lot of time (as shown in Table 3), while LC-VGG-B does not.

We also compare the speeds of other current methods in Table 3. It can be found that our method can work well without using additional network and improve efficiency without sacrificing accuracy.

Performance on irregular datasets. Irregular datasets include IC15, SVTP and CUTE. Due to the arbitrariness of irregular text shapes, it is necessary to use the local correlation of adjacent pixels for recognition. As shown in Table 2, Our method LC-VGG-B achieves the best performance to the existing methods on all irregular datasets, except for Cheng et al. [8] on CUTE. In further, we are 3.9% better than the state-of-the-art method [7] on dataset IC15. For method [8], it encodes the features in four directions and uses 90k+ST$^+$ as train dataset, while our method encode in one direction and use 90k+ST which is smaller

Methods	training speed(fps)	inference speed(fps)
Cheng et al.[7]	90	-
Shi et al. [2]	160	5
Cheng et al.[8]	190	630*
Shi et al. [36]	360	50
Ours	370	95

Table 3: Speed of different methods. fps means the number of images per second. '*' means unknown batchsize.

than 90k+ST$^+$. But our method still performs better than [8] on IC15 and SVTP. We also compare our method LC-VGG-A with Shi ei al. [36] with VGG-net on the irregular datasets. As shown in Table 1, on the condition of the same network bone and train data, LC-VGG-A is better than [36] with VGG-net [36] on IC15 and CUTE.

As a whole, our method has achieved good results in both the irregular scene text and the regular scene text recognition by using less time.

5 Conclusion

Deep convolutional neural networks (DCNN) become the dominant approach for many tasks, such as scene text recognition and object detection, etc. In the traditional CNNs, the filter describes a fixed of patterns in images. In order to extract much richer feature representations, lots of architectures have been proposed building upon convolutional layers or designing much deeper networks, which needs additional computing and storage resources. Feature vectors which belong to the same image text region have similar semantics and are close to each other. In this paper, we present an LC mechanism to encode the current feature vector by using the local correlation between neighbor vectors. This mechanism can be easily embedded in other architectures used in computer vision tasks. By experimenting on seven datasets, the method we proposed is superior to other existing methods in the comprehensive performance of speed and accuracy. We plan to extend our approach to other computer vision tasks such as image classification etc.

References

[1] K. Cho B. Dzmitry and et al. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv*, 1409.0473, 2014.

[2] X. Wang B. Shi and et al. Robust scene text recognition with automatic rectification. *CVPR*, 2016.

[3] C. Bartz, H. Yang, and et al. See: Towards semi-supervised end-to-end scene text recognition. *in AAAI*, 1, 2018.

[4] A. Bissacco, M. Cummins, and et al. Photoocr: Reading text in uncontrolled conditions. *in ICCV*, pages 785–792, 2013.

[5] A. Buades, B. Coll, and et al. A non-local algorithm for image denoising. *in CVPR*, pages 60–65, 2005.

[6] L. Chen-Yu and S. Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016.

[7] Z. Cheng, F. Bai, and et al. Focusing attention: Towards accurate text recognition in natural images. *in ICCV*, pages 5086–5094, 2017.

[8] Z. Cheng, Y. Xu, and et al. Aon: Towards arbitrarily-oriented text recognition. *in CVPR*, pages 5571–5579, 2018.

[9] Y. Gao, Y. Chen, J. Wang, and et al. Reading scene text with attention convolutional sequence modeling. *arXiv preprint arXiv*, 1709.04303, 2017.

[10] J. Gehring, M. Auli, and et al. Convolutional sequence to sequence learning. *arXiv preprint arXiv*, 1705.0312, 2017.

[11] A. Graves, S. Fernandez, and et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. *in ICML*, 2006.

[12] A. Gupta, A. Vedaldi, and et al. Synthetic data for text localisation in natural images. *CVPR*, page 2315–2324, 2016.

[13] P. He, W. Huang, and et al. Reading scene text in deep convolutional sequences. *in AAAI*, pages 3501–3508, 2016.

[14] T. He, Z. Tian, and et al. An end-to-end textspotter with explicit alignment and attention. *in AAAI*, 2018.

[15] M. Jaderberg, K. Simonyan, and et al. Deep structured output learning for unconstrained text recognition. *arXiv preprint arXiv*, 1412.5903, 2014.

[16] M. Jaderberg, A. Vedaldi, and et al. Deep features for text spotting. *in ECCV*, pages 512–528, 2014.

[17] M. Jaderberg, K. Simonyan, and et al. Spatial transformer networks. *in NIPS*, page 2017–2025, 2015.

[18] D. Karatzas, F. Shafait, and et al. Icdar 2013 robust reading competition. *in ICDAR*, 2013.

[19] D. Karatzas, L. Gomez-Bigorda, and et al. Icdar 2015 competition on robust reading. *in ICDAR*, pages 1156–1160, 2015.

[20] D.P Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv*, 1412.6980, 2014.

[21] M. Krystian and C. Schmid. Scale affine invariant interest point detectors. *International journal of computer vision*, 60(1):971–987, 2004.

[22] W. Liu, C. Chen, and et al. Char-net: A character-aware neural network for distorted scene text recognition. *in AAAI*, 2018.

[23] X. Liu, D. Liang, and et al. Recursive recurrent nets with attention modeling for ocr in the wild. *in CVPR*, pages 2231–2239, 2018.

[24] S. M. Lucas, A. Panaretos, and et al. Icdar 2003 robust reading competitions: entries, results, and future directions. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3):105–122, 2005.

[25] Minh-Thang Luong, H. Pham, and et al. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv*, 1508.04025, 2015.

[26] B. Mandelbrot. How long is the coast of britain? statistical self-similarity and fractional. *Science*, 156:636–638, 1967.

[27] J. Max, K. Simonyan, and et al. Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, 116(1):1–20, 2016.

[28] A. Mishra, K. Alahari, and et al. Scene text recognition using higher order language priors. *in BMVC*, 2012.

[29] A. Mishra, K. Alahari, and et al. Top-down and bottom-up cues for scene text recognition. *in CVPR*, pages 2687–2694, 2012.

[30] T. Q. Phan, P. Shivakumara, and et al. A gradient vector flow-based method for video character segmentation. *in ICDAR*, pages 1024–1028, 2011.

[31] T. Q. Phan, P. Shivakumara, and et al. Recognizing text with perspective distortion in natural scenes. *in ICCV*, 2013.

[32] A. Risnumawan, P. Shivakumara, and et al. A robust arbitrary text detection system for natural scene images. *Expert Syst. Appl*, 41(18):8027—-8048, 2014.

[33] H. Xie S. Fang and et al. Attention and language ensemble for scene text recognition with convolutional sequence modeling. *in MM*, 2018.

[34] B. Shi, X. Wang, and et al. Robust scene text recognition with automatic rectification. *in CVPR*, pages 4168–4176, 2016.

[35] B. Shi, X. Bai, and et al. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE PAMI*, 39.11: 2298–2304, 2017.

[36] B. Shi, M. Yang, and et al. Aster: An attentional scene text recognizer with flexible rectification. *IEEE PAMI*, 2018.

[37] Karen Simonyan, Zisserman, and Andrew. Very deep convolutional networks for large-scale image recognition. *Computer Science*, 2014.

[38] A. Vaswani, N. Shazeer, and et al. Attention is all you need. *in NIPS*, 2017.

[39] J. Wang and X. Hu. Gated recurrent convolution neural network for ocr. *in NIPS*, 334-343, 2017.

[40] K. Wang, B. Babenko, and et al. End-to-end scene text recognition. *in ICCV*, 2011.

[41] T. Wang, D. J.Wu, and et al. End-to-end text recognition with convolutional neural networks. *in ICPR*, 2012.

[42] X. Wang, R. Girshick, and et al. Non-local neural networks. *arXiv preprint arXiv*, 1711.07971, 2017.

[43] X. Yang, D. He, and et al. Learning to read irregular text with attention mechanisms. *in IJCAI*, pages 3280—3286, 2017.

[44] A. Fan Yann N. Dauphinand and et al. Language modeling with gated convolutional networks. *in ICML*, 2017.

[45] C. Yao, X. Bai, and et al. Strokelets: A learned multi-scale representation for scene text recognition. *in CVPR*, pages 4042–4049, 2014.