Arbitrarily-Oriented Text Recognition

Zhanzhan Cheng[1] Yangliu Xu[12] Fan Bai[3]
Yi Niu[1] Shiliang Pu[1] Shuigeng Zhou[3]*
[1]Hikvision Research Institute, China; [2]Tongji University, China;
[3]Shanghai Key Lab of Intelligent Information Processing and
School of Computer Science, Fudan University, Shanghai, China
{chengzhanzhan;xuyangliu;niuyi;pushiliang}@hikvision.com;
{baif13;sgzhou}@fudan.edu.cn

Abstract

Recognizing text from natural images is still a hot research topic in computer vision due to its various applications. Despite the enduring research of several decades on optical character recognition (OCR), recognizing texts from natural images is still a challenging task. This is because scene texts are often in irregular arrangements (curved, arbitrarily-oriented or seriously distorted), which have not yet been well addressed in the literature. Existing methods on text recognition mainly work with regular (horizontal and frontal) texts and cannot be trivially generalized to handle irregular texts. In this paper, we develop the arbitrary orientation network (AON) to capture the deep features of irregular texts (e.g. arbitrarily-oriented, perspective or curved), which are combined into an attention-based decoder to generate character sequence. The whole network can be trained end-to-end by using only images and word-level labels. Extensive experiments on various benchmarks, including the CUTE80, SVT-Perspective, IIIT5k, SVT and ICDAR datasets, show that the proposed AON-based method substantially outperforms the existing methods.

Figure 1: Examples of irregular (slant/perspective, curved and oriented etc.) texts in natural images. Subfigures (a) - (b), (c) - (d) and (e) - (f) are slant/perspective, curved and oriented images respectively.

Introduction

Scene text recognition has attracted much research interest of the computer vision community (Neumann and Matas 2012; Jaderberg et al. 2014; Lee and Osindero 2016; Shi, Bai, and Yao 2016; Yang et al. 2017; Cheng et al. 2017) because of its great help to various applications such as road sign recognition and navigation reading for advanced driver assistant system (ADAS). Though Optical Character Recognition (OCR) has been extensively studied for several decades, recognizing texts from natural images is still a challenging task due to complicated environments (e.g. uneven lighting, burring, perspective distortion and orientation).

Motivation. In the past years, there have been many works to solve scene text recognition (Lee and Osindero 2016; Shi, Bai, and Yao 2016; Yang et al. 2017; Cheng et al. 2017). Although these approaches have shown promising results, most of them can effectively handle only regular texts that are often tightly-bounded, horizontal and frontal. However, in real-world applications, many scene texts are in irregular arrangements (e.g. arbitrarily-oriented, curved, slant and perspective etc.) as shown in Fig. 1, so most existing methods cannot be widely applied in practice.

Recently, there are two related works aiming at irregular texts: the spatial transformer network (STN) (Jaderberg et al. 2015b) - based method by (Shi et al. 2016) and the attention-based method with fully convolutional network (FCN) (Long, Shelhamer, and Darrell 2015) by (Yang et al. 2017). (Shi et al. 2016) attempted to first rectify irregular (e.g. curved or perspectively distorted) texts to approximately regular texts, then recognized the rectified images with an attention-based sequence recognition network. However, in complicated (e.g. arbitrarily-oriented or serious curved) natural scenes, it is hard to optimize the STN-based method without human-labeled geometric ground truth. Besides, training STN needs sophisticated skills. For example, the thin-plate-spline (TPS) (Bookstein 1989)-based STN (Shi et al. 2016) should be given some initialization pattern for the fiducial points, and is not quite effective for arbitrarily-oriented scene texts. (Yang et al. 2017) introduced an auxiliary dense character detection task for encouraging the learning of visual representations with a fully convolutional network. Though the method showed better performance on irregular texts, it was carried out with an exhausting multi-task learning (MTL) strategy and relied on

* Corresponding author.

character-level bounding box annotations. Note that, though the attention-based model has the potential to perform 2D feature selection (Xu et al. 2015), we found in experiments that directly training attention-based model on irregular texts is difficult due to irregular character placements. This situation motivates us to explore new and more effective methods to recognize irregular scene texts.

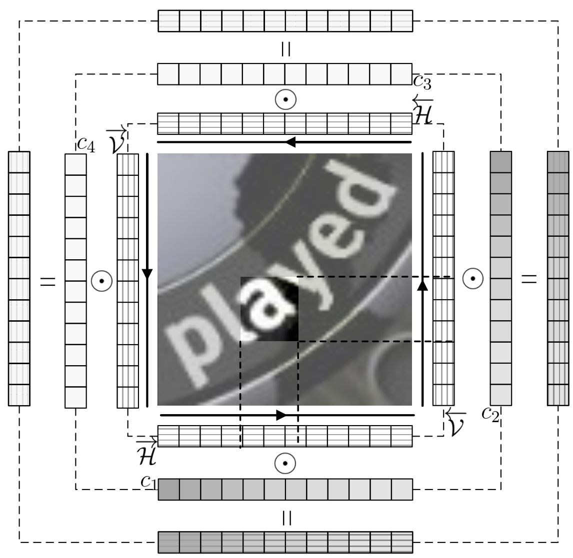

Figure 2: Illustration of character visual representation in four directions: $\overrightarrow{\mathcal{H}}$: $left \rightarrow right$, $\overleftarrow{\mathcal{H}}$:$right \rightarrow left$, $\overrightarrow{\mathcal{V}}$:$top \rightarrow bottom$ and $\overleftarrow{\mathcal{V}}$:$bottom \rightarrow top$ and four character placement clues c_1, c_2, c_3 and c_4. Here, there are three squares connected with dashed lines. The innermost square represents the four 1D sequences of features, each comes along with an arrowed line. The middle square refers to the placement clues used for weighting the corresponding sequences of features. The outermost square stands for the weighted feature sequence by conducting Hadamard product $\odot$ with character placement clues and horizontal/vertical features. For the character 'a' in the image, we can represent it by the four weighted sequences of features.

Basic Idea. From the above analysis, we can see that most existing methods directly encode a text image as a 1D sequence of features and then decode them to the predicted text, which implies that any text in an image is treated in the same direction such as *from left to right* by default. However, this is not true in the wild. After carefully analyzing the typical character placement styles of natural text images, we suggest that the visual representation of an arbitrarily-oriented character in a 2D image can be described in four directions: $left \rightarrow right$, $right \rightarrow left$, $top \rightarrow bottom$ and $bottom \rightarrow top$. Concretely, we can encode the input image to four feature sequences of four directions: *horizontal features* ($\overrightarrow{\mathcal{H}}$), *reversed horizontal features* ($\overleftarrow{\mathcal{H}}$), *vertical features* ($\overrightarrow{\mathcal{V}}$) and *reversed vertical features* ($\overleftarrow{\mathcal{V}}$), as shown in Fig. 2, and the length of each sequence is equal. The horizontal/vertical features can be extracted by downsampling the height/width of feature maps to 1. In order to represent an arbitrarily-oriented character, a weighting mechanism can be used to combine the four feature sequences of different directions. We call the weights *character placement clues*, which are denoted as c_1, c_2, c_3 and c_4 in Fig. 2. The character placement clues can be learned from the input images with a convolutional-based network, which guides to effectively integrate the four sequences of features, and then a filter gate (FG) generates the integrated feature sequence as the character's visual representation. Therefore, an arbitrarily-oriented character in a 2D image can be represented as the combination of horizontal and vertical features by conducting the Hadamard product with the sequences of features and the corresponding placement clues. In Fig. 2, c_1 and c_2 play the dominant role in determining the visual representation of character 'a'. In this paper, we call the four-direction feature extraction network and the clues extraction network *arbitrary orientation network* (AON), which means that it can effectively handle arbitrarily-oriented texts.

Contributions. In this paper, we develop a novel method for robustly recognizing both regular and irregular natural texts by employing the proposed *arbitrary orientation network* (AON). Major contributions of this paper are as follows:

1. We propose the arbitrary orientation network (AON) to extract scene text features in four directions and the character placement clues.
2. We design a filter gate (FG) for fusing four-direction features with the learned placement clues. That is, FG is responsible for generating the integrated feature sequence.
3. We integrate AON, FG and an attention-based decoder into the character recognition framework. The whole network can be directly trained end-to-end without any character-level bounding box annotations.
4. We conduct extensive experiments on several public irregular and regular text benchmarks, which show that our method substantially outperforms the existing ones.

Related works

In recent years, several methods have been proposed for scene text recognition. For the general information of text recognition, readers can refer to Ye and Doermann's recent survey (Ye and Doermann 2015). Basically, there are two types of scene text recognition approaches: bottom-up and top-down.

Traditional methods mostly follow the *bottom-up* pipeline: first extracting low-level features for individual character detection and recognition one by one, then integrating these characters into words based on a set of heuristic rules or a language model. For example, (Neumann and Matas 2012) defined a set of handcrafted features such as aspect ratio, hole area ratio etc. to train a Support Vector Machine (SVM) classifier. (Wang and Belongie 2010; Wang, Babenko, and Belongie 2011) first fetched each character in the cropped word image by sliding window, then recognized it with a character classifier trained by the extracted HOG descriptors (Yao et al. 2014). However, the performance of these methods is limited due to the low representation capability of handcrafted features. With the advancement of neural-network-based methods, many researchers developed deep neural architectures and achieved better results. (Bissacco et al. 2013) adopted a fully connected network of 5 hidden layers for character feature representation, then used an n-gram language model to recognize characters. (Wang et al. 2012) developed a CNN-based feature extraction framework for character recognition, and applied a non-maximum suppression method for final word predictions. (Jaderberg et al. 2015a) also proposed a CNN-based method with structured output layer for unconstrained recognition. These above methods require the segmentation of each character, which can be very challenging because of the complicated background clutter and the inadequate distance between consecutive characters. Besides, segmentation annotations require additional resource consuming.

The other approaches work in a top-down style: directly predicting the entire text from the original image without detecting the characters. (Jaderberg et al. 2016) conducted a 90k-class classification task with a CNN, in which each class represents an English word. Consequently, the model can not recognize out-of-vocabulary words. Recent works solve this problem as a sequence recognition problem, where images and texts are separately encoded as patch sequences and character sequences, respectively. (Sutskever, Vinyals, and Le 2014) extracted sequences of HOG features to represent images, and generated the character sequence with the recurrent neural network (RNN). (He et al. 2016b) and (Shi, Bai, and Yao 2016) proposed the end-to-end neural networks that combines CNN and RNN for visual feature representation, then the CTC (Graves et al. 2006) Loss was combined with the RNN outputs for calculating the conditional probability between the predicted and the target sequences. (Lee and Osindero 2016) used a recursive CNN to learn broader contextual information, and applied the attention-based decoder for sequence generation. (Cheng et al. 2017) proposed a focus mechanism to eliminate the attention drift to improve the regular text recognition performance. However, since a text image is encoded into a 1D-based sequence of features, these methods can not effectively handle the irregular texts such as the arbitrarily-oriented texts. In order to recognize irregular texts, (Shi et al. 2016) applied the spatial transformer network (STN) (Jaderberg et al. 2015b) for text rectification, then recognized the rectified text images with the sequence recognition network. (Yang et al. 2017) introduced an auxiliary dense character detection task for encouraging the learning of visual representations with a fully convolutional network (FCN) (Long, Shelhamer, and Darrell 2015). In practice, training STN-based methods is extremely difficult without human-labeled geometric ground truth, especially for texts in complicated (e.g. curved, arbitrarily-oriented or perspective etc.) environments. Besides, sophisticated tricks are also required. For example, to train the thin-plate-spline (TPS) (Bookstein 1989)-based STN (Shi et al. 2016)-based method, the initialization pattern should be given for the fiducial points. Though (Yang et al. 2017) can recognize characters in a 2D image, the method relies on the multi-task learning framework (including 3 task branches and 2 tunable super-parameters) and character-level bounding box annotations, which results in large amount of resource consuming. While obtaining better performance on irregular texts, it performs worse on regular texts.

Different from existing approaches, in this paper we first extract deep feature representations of images by using an arbitrary orientation network (AON), then use a filter gate (FG) to generate the integrated sequence of features, which are fed to an attention-based decoder for generating predicted sequences. Furthermore, we can once for all train the whole network end-to-end with only word-level annotations.

Note that in the OCR field, natural text reading systems often consist of two steps: 1) detecting each word's location in natural images and 2) recognizing text from the cropped image. In general, robust detection is helpful in recognizing texts. Therefore, several methods (Zhang et al. 2016; Shi, Bai, and Belongie 2017; Ma et al. 2017; Jiang et al. 2017) have been proposed for multi-oriented text detection. Though this work focuses on the recognition task, our AON-based method can directly recognize arbitrarily-oriented texts, which alleviates the pressure of text detection.

Architecture

The whole network architecture is shown in Fig. 3, which consists of four major components: 1) The *basal convolutional neural network* (BCNN) for extracting low-level visual features, 2) the *arbitrary orientation network* (AON) for generating four-direction sequences of features and the character placement clues, 3) the *filter gate* (FG) for combining the four sequences of features with the learned placement clues to generate the integrated feature sequence, and 4) the *attention-based decoder* for predicting character sequence.

Basal Convolutional Neural Network (BCNN)

The BCNN module is responsible for capturing the foundational visual representation of text images, and outputs a group of feature maps. As shown in Fig. 3, we use four convolution blocks as the foundational feature extractor. The outputs of BCNN must be square feature maps.

Multi-Direction Feature Extraction Module

This module includes the arbitrary orientation network (AON) and the filter gate (FG), which constitute the core of the proposed method. With the extracted foundational features, we devise AON for capturing arbitrarily-oriented text features and the corresponding character placement clues. We also design FG for integrating multi-direction features by using the character placement clues. The details of AON and FG will be described in next section.

Attention-based Decoder

An attention-based decoder is a recurrent neural network (RNN) that directly generates the target sequence $(y_1, ..., y_M)$ from an input feature sequence $(\hat{h}_1, ..., \hat{h}_L)$. Bahdanau et al. (Bahdanau, Cho, and Bengio 2015) first proposed the architecture of attention-based decoder. At the t-th step, the attention module generates an output y_t as follows:

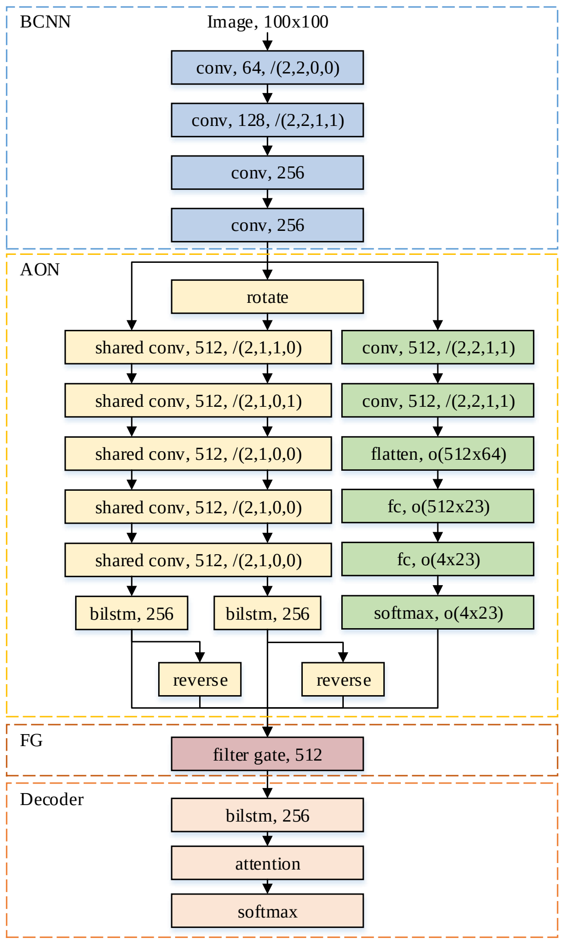

Figure 3: The network architecture of our method, which consists of four components: 1) the basal convolutional neural network (BCNN) module for low-level visual representation; 2) the arbitrary orientation network (AON) for capturing the horizontal, vertical and character placement features; 3) the filter gate (FG) for combing four feature sequences with the character placement clues; 4) the attention-based decoder (Decoder) for predicting character sequence. The above four modules are shown in the blue, golden, dull-red and brown dashed boxes, respectively. Meanwhile, all convolution or shared convolution blocks have the following format: $name, c[, /(s_h, s_w, p_h, p_w)]$. The bilstm and filter gate blocks are represented as $name, c$. The flatten, fc (fully-connected) and softmax operations have the format: $name, o(c, l)$. Here, c, s_h, s_w, p_h, p_w, l, $/$ and o represent the number of channels, stride height, stride width, pad height, pad width, length of feature maps, pooling operation and output shape, respectively. The whole network can be trained end-to-end.

$$y_t = softmax(W^T s_t), \tag{1}$$

where W^T is a learnable parameter, and s_t is the RNN hidden state at time t, computed by

$$s_t = RNN(y_{t-1}, g_t, s_{t-1}), \tag{2}$$

where g_t is the weighted sum of sequential feature vectors $\hat{\mathcal{H}} : (\hat{h}_1, ..., \hat{h}_L)$, that is,

$$g_t = \sum_{j=1}^{L} \alpha_{t,j} \hat{h}_j, \tag{3}$$

where $\alpha_t \in \mathbb{R}^L$ is a vector of the *attention weights*, also called *alignment factors* (Bahdanau, Cho, and Bengio 2015). In the computation of *attention weights*, α_t is often evaluated by scoring each element in $\hat{\mathcal{H}}$ separately and normalizing the scores as follows:

$$\alpha_t = Attend(s_{t-1}, \hat{\mathcal{H}}), \tag{4}$$

where $Attend$ describes the attending process (Chorowski et al. 2015).

Above, the RNN function in Eq. (2) represents an LSTM recurrent network. Note that the decoder is capable of generating sequences of variable lengths. Following (Sutskever, Vinyals, and Le 2014), a special end-of-sequence (EOS) token is added to the target set, so that the decoder completes the generation of characters when EOS is emitted.

Network Training

We integrate the BCNN, AON, FG and attention decoder into one network, as shown in Fig. 3. Therefore, given an input image $\mathcal{I}$, the loss function of the network is as follows:

$$\mathcal{L} = -\sum_t lnP(\hat{y}_t | \mathcal{I}, \theta), \tag{5}$$

where $\hat{y}_t$ is the ground truth of the t-th character and θ is a vector that combines all the network parameters.

Character Sequence Decoding

In general, decoding is the final process to generate the predicted characters. Following the decoding conventions, two processing modes are given: unconstrained (lexicon-free) mode and constrained mode. We execute unconstrained text recognition by directly selecting the most probable character. While in constrained text recognition, with respect to different types of lexicons (their sizes are denoted by "50", "1k" and "full" respectively), we calculate the conditional probability distributions for all lexicon words, and take the one with the highest probability as the output result.

Technical Details of AON and FG

Arbitrary Orientation Network (AON)

We develop an *arbitrary orientation network* (AON) consisting of the *horizontal network* (HN), the *vertical network* (VN) and the *character placement clue network* (CN) for extracting horizontal, vertical and placement features respectively.

The HN encodes the foundational feature maps into a sequence of horizontal feature vectors $\mathcal{H} \in \mathbb{R}^{L\times D}$ by first performing downsampling on height directly by 5 shared convolutional blocks (described bellow) with the corresponding pooling strategy (shown in Fig. 3) to 1, and using the bidirectional LSTM to further encode the feature sequence, then generating the reversed feature sequence by conducting reverse operation (described in Eq. (6) and (7)), where L and D represent the length of $\hat{\mathcal{H}}$ and the channel number, respectively. Symmetrically, VN first rotates the square feature maps by 90 degrees, then generates the vertical feature vectors $\mathcal{V} \in \mathbb{R}^{L\times D}$ with the same procedure as HN.

Since we describe each character sequence in four directions: $left \rightarrow right$, $right \rightarrow left$, $top \rightarrow bottom$ and $bottom \rightarrow top$, $\mathcal{H}$ and $\mathcal{V}$ can be represented as follows:

$$\mathcal{H} = \begin{cases} \overrightarrow{\mathcal{H}} : (h_1, ..., h_L)^T, & left \rightarrow right \\ \overleftarrow{\mathcal{H}} : (h_L, ..., h_1)^T, & right \rightarrow left \end{cases} \tag{6}$$

$$\mathcal{V} = \begin{cases} \overrightarrow{\mathcal{V}} : (v_1, ..., v_L)^T, & top \rightarrow bottom \\ \overleftarrow{\mathcal{V}} : (v_L, ..., v_1)^T. & bottom \rightarrow top \end{cases} \tag{7}$$

For each text image, the CN outputs the corresponding character placement clues $\mathcal{C} \in \mathbb{R}^{4\times L}$ as:

$$\mathcal{C} = (c_1, ..., c_L)^T. \tag{8}$$

Here, for any $c_i \in \mathbb{R}^4$, we have $\sum_{j=1}^{4} c_{ij} = 1$, where c_{ij} refers to the j-th direction's weight. The extraction process of clues is depicted as the green blocks in Fig. 3.

In practice, we find that it is hard to train the HN and VN respectively and simultaneously. The state of each branch is easy to corrupted on orientation distribution unbalanced training datasets. Therefore, we design a shared convolution mechanism that performs the same convolutional filter operations for both horizontal and vertical process, and the shared convolution block is shown in Fig. 4. With the shared convolutional mechanism, the network is robust and easy to learn on orientation unbalanced training datasets.

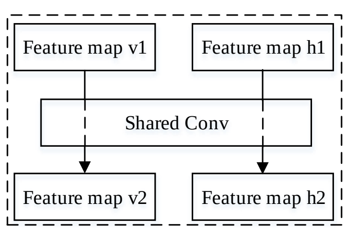

Figure 4: The shared convolution block in the dashed box provides a mechanism that multi-groups of feature maps share the same convolutional filters.

Filter Gate (FG)

With the captured four feature sequences and character placement clues, we design a filter gate to neglect the irrelevant features. Formally, given the i-th features $(\overrightarrow{\mathcal{H}}_i, \overleftarrow{\mathcal{H}}_i, \overrightarrow{\mathcal{V}}_i, \overleftarrow{\mathcal{V}}_i)$, we use the corresponding placement clue c_i to attend the appropriate features:

$$\hat{h}'_i = [\overrightarrow{\mathcal{H}}_i \ \overleftarrow{\mathcal{H}}_i \ \overrightarrow{\mathcal{V}}_i \ \overleftarrow{\mathcal{V}}_i] c_i. \tag{9}$$

Then an activation operation is performed as follows:

$$\hat{h}_i = \tanh(\hat{h}'_i). \tag{10}$$

Above, $\hat{h}_i$ indicates the i-th element of $\hat{\mathcal{H}} : (\hat{h}_1, ..., \hat{h}_L)$.

Performance Evaluation

We conduct extensive experiments to validate the effectiveness of the proposed method on both irregular and regular recognition benchmarks. We all know that texts in the wild are usually irregular. First, we test our model on three irregular benchmarks with many arbitrarily-oriented, curved and perspective texts. Then, we evaluate our method on three general datasets, which mainly consist of frontal and horizontal texts. At last, we discuss the effectiveness of character placement clues over the input origin images.

Datasets

The regular and irregular benchmarks are as follows:

SVT-Perspective (Quy Phan et al. 2013) contains 639 cropped images for testing. Images are picked from side-view angle snapshots in Google Street View, therefore one may observe severe perspective distortions. Each image is associated with a 50-word lexicon and a full lexicon.

CUTE80 (Risnumawan et al. 2014) is specifically collected for evaluating the performance of curved text recognition. It contains 288 cropped natural images for testing. No lexicon is associated.

ICDAR 2015 (IC15 in short) (Karatzas et al. 2015) contains 2077 cropped images including more than 200 irregular (arbitrarily-oriented, perspective or curved) images. No lexicon is associated.

IIIT5K-Words (IIIT5K in short) (Mishra, Alahari, and Jawahar 2012) is collected from the Internet, containing 3000 cropped word images in its test set. Each image specifies a 50-word lexicon and a 1k-word lexicon, both of which contain the ground truth words as well as other randomly picked words.

Street View Text (SVT in short) (Wang, Babenko, and Belongie 2011) is collected from the Google Street View, consists of 647 word images in its test set. Many images are severely corrupted by noise and blur, or have very low resolutions. Each image is associated with a 50-word lexicon.

ICDAR 2003 (IC03 in short) (Lucas et al. 2003) contains 251 scene images, labeled with text bounding boxes. Each image is associated with a 50-word lexicon defined by Wang et al. (Wang, Babenko, and Belongie 2011). For fair comparison, we discard images that contain non-alphanumeric characters or have less than three characters, following (Wang, Babenko, and Belongie 2011). The resulting dataset contains 867 cropped images. The lexicons include the 50-word lexicons and the full lexicon that combines all lexicon words.

Table 1: Results on irregular benchmarks. “50” is lexicon size and “Full” indicates the combined lexicon of all images in the benchmarks. “None” means lexicon-free.

Method	SVT-Perspective			CUTE80	IC15
	50	Full	None	None	None
ABBYY (Wang, Babenko, and Belongie 2011)	40.5	26.1	–	–	–
Mishra (Graves, r. Mohamed, and Hinton 2013)	45.7	24.7	–	–	–
Wang (Wang et al. 2012)	40.2	32.4	–	–	–
Phan (Quy Phan et al. 2013)	75.6	67.0	–	–	–
Shi (Shi, Bai, and Yao 2016)	92.6	72.6	66.8	54.9	–
Shi (Shi et al. 2016)	91.2	77.4	71.8	59.2	–
Yang (Yang et al. 2017)	93.0	80.2	**75.8**	69.3	–
Cheng (Cheng et al. 2017)	92.6	81.6	71.5	63.9	66.2
Ours	**94.0**	**83.7**	73.0	**76.8**	**68.2**

Implementation Details

Network details: The deep neural network has been detailed in Fig. 3. In our network, all images are resized to 100×100. As for the convolutional strategy, all convolutional blocks have 3×3 size of kernels, 1×1 size of pads and 1×1 size of strides, and all pooling (max) blocks have 2×2 size of kernels. We adopt batch normalization (BN) (Ioffe and Szegedy 2015) and ReLU activation right after each convolution. For the character generation task, the attention is designed with an LSTM (256 memory blocks) and 37 output units (26 letters, 10 digits, and 1 EOS symbol).

Implementation and Running Environment: We train our model on 8-million synthetic data released by Jaderberg et al. (Jaderberg et al. 2014) and 4-million synthetic instances (excluding the images that contain non-alphanumeric characters) cropped from 80-thousand images released by (Gupta, Vedaldi, and Zisserman 2016) by the ADADELTA (Zeiler 2012) optimization method. Meanwhile, we conduct data augmentation by randomly rotating each image range from $0°$ to $360°$ once. Our method is implemented under the Caffe framework (Jia et al. 2014). We use the CUDA 8.0 and CUDNN v7 backend extensively in our implementation, so that most modules in our method are GPU-accelerated. Our experiments are carried out on a workstation with one Intel Xeon(R) E5-2650 2.30GHz CPU, one NVIDIA Tesla P40 GPU, and 128GB RAM.

Performance on Irregular Datasets

Tab. 1 summarizes the recognition results on three irregular text datasets: SVT-Perspective, CUTE80 and ICDAR15. For (Cheng et al. 2017), we test FAN on irregular datasets. We find that our method outperforms the existing methods on almost all benchmarks, but SVT-Perspective with lexicon-free released by (Yang et al. 2017). However, (Yang et al. 2017) implicated its text-reading system with both word-level and character-level bounding box annotations, which is time and money consuming. Our method can be easily carried out with only the word-level annotations. Fig. 5 shows some examples recognized by our method, in which the text-reading orientations are also depicted with the character placement clues. We will discuss that in detail later.

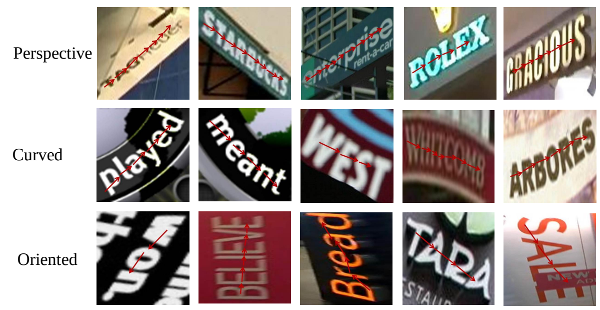

Figure 5: The visualization of character placement clues over perspective, curved and oriented images, shown in the first, second and last line, respectively. The red arrows refer to the text orientations, with the tails and heads represent the last and current character positions, respectively. All texts in the images are corrected predicted by our method.

Performance on Regular Datasets

We also test our method on three regular text datasets shown in Tab. 2 to verify the performance of our method in general scene. Since (Jaderberg et al. 2016) treated each word as a class label and conducted the classification task and their model cannot recognize out-of-vocabulary words. We do not do comparison with the results of (Jaderberg et al. 2016). In the unconstrained cases, our method achieves the best results on IIIT5K, SVT and IC03. In the constrained cases, our method performs best on IIIT5K, but falls behind (Shi, Bai, and Yao 2016) slightly on SVT and IC03 and (Yang et al. 2017) on IC03. However, (Shi, Bai, and Yao 2016) encodes an image into a 1D sequence, therefore irregular text with arbitrarily-oriented or curved angle cannot be successfully recognized. Note that though (Cheng et al. 2017) designed a 32-layer ResNet (He et al. 2016a) for extracting robust features, our method still outperforms the baseline results of (Cheng et al. 2017) on all benchmarks. In conclusion, the proposed method is capable of effectively recognizing both regular and irregular texts.

Table 2: Results on regular benchmarks. “50” and “1k” are lexicon sizes. “Full” indicates the combined lexicon of all images in the benchmarks. “None” means lexicon-free.

Method	IIIT5k			SVT		IC03		
	50	1k	None	50	None	50	Full	None
ABBYY (Wang, Babenko, and Belongie 2011)	24.3	–	–	35.0	–	56.0	55.0	–
Wang (Wang, Babenko, and Belongie 2011)	–	–	–	57.0	–	76.0	62.0	–
Mishra (Graves, r. Mohamed, and Hinton 2013)	64.1	57.5	–	73.2	–	81.8	67.8	–
Wang citewang2012end	–	–	–	70.0	–	90.0	84.0	–
Goel (Goel et al. 2013)	–	–	–	77.3	–	89.7	–	–
Bissacco (Bissacco et al. 2013)	–	–	–	90.4	78.0	–	–	–
Alsharif (Alsharif and Pineau 2014)	–	–	–	74.3	–	93.1	88.6	–
Almazán (Almazán et al. 2014)	91.2	82.1	–	89.2	–	–	–	–
Yao (Yao et al. 2014)	80.2	69.3	–	75.9	–	88.5	80.3	–
Jaderberg (Jaderberg et al. 2015a)	–	–	–	86.1	–	96.2	91.5	–
Su (Su and Lu 2015)	–	–	–	83.0	–	92.0	82.0	–
Gordo (Gordo 2015)	93.3	86.6	–	91.8	–	–	–	–
Jaderberg (Jaderberg et al. 2015a)	95.5	89.6	–	93.2	71.7	97.8	97.0	89.6
Shi (Shi, Bai, and Yao 2016)	97.6	94.4	78.2	**96.4**	80.8	**98.7**	97.6	89.4
Shi (Shi et al. 2016)	96.2	93.8	81.9	95.5	81.9	98.3	96.2	90.1
Lee (Lee and Osindero 2016)	96.8	94.4	78.4	96.3	80.7	97.9	97.0	88.7
Yang (Yang et al. 2017)	97.8	96.1	–	95.2	–	–	**97.7**	–
Cheng (Cheng et al. 2017)	98.9	96.8	83.7	95.7	82.2	98.5	96.7	91.5
Ours	**99.6**	**98.1**	**87.0**	96.0	**82.8**	98.5	97.1	**91.5**

Visualization of Character Placement Clues

In this section, we explore the effectiveness of character placement clues by locating each character and painting text orientations in the origin images.

We all know that the alignment factors α_t produced by the attention module indicate the probability distributions over the input sequence of features for generating the glimpse vector g_t. And the four character placement clues $\mathcal{C} = [c_1, c_2, c_3, c_4]$ imply the importance of four extracted feature sequences for representing characters. With $\mathcal{C}$ and α_t, we roughly divide the input image into $L \times L$ patches and calculate the character position distribution dis by

$$dis = \mathcal{C} \odot \alpha_t, \tag{11}$$

where $dis = (d_1, d_2, d_3, d_4) \in \mathbb{R}^{4\times L}$. We further normalize each element by

$$norm(d_{ij}) = \frac{d_{ij}}{\sum_{i=1}^{2}\sum_{j=1}^{L} d_{ij}}, \tag{12}$$

for $i \in (1, 2)$, and by

$$norm(d_{ij}) = \frac{d_{ij}}{\sum_{i=3}^{4}\sum_{j=1}^{L} d_{ij}}, \tag{13}$$

for $i \in (3, 4)$, where $norm$ indicates the normalization operation.

For a character at position (x, y), we first compute the horizontal coordinate x with $[d_1, d_2]$ by

$$x = \sum_{j=1}^{L}\sum_{i=1}^{2} j \times norm(d_{ij}), \tag{14}$$

where $i \in (1, 2)$ and $j \in (1, 2, ..., L)$. Similarity, we compute the vertical coordinate y with $[d_3, d_4]$ by

$$y = \sum_{j=1}^{L}\sum_{i=3}^{4} j \times norm(d_{ij}), \tag{15}$$

To visualize character placement clues on the input images, we mark the corresponding coordinate (x, y) on the input images as the generated characters' position one by one, and consecutively connect the last character's position and the current character's position with an arrow to describe the text's orientation. Fig. 5 shows the position of each character and the text orientations in the input images. We can see that the curves form the connected arrows basically conform to the orientations of the texts, which shows that our method is effective in estimating the orientations of texts in images.

Conclusion

In this work, we propose a novel method to recognize arbitrarily oriented texts by 1) devising an arbitrary orientation network to extract visual features of characters in four directions and the character placement clues, 2) using a filter gate mechanism to combine the four-direction sequences of features, and 3) employing an attention-based decoder for generating character sequence. Different from the existing methods, our method can effectively recognize both irregular and regular texts from images. Experiments over both regular and irregular benchmarks validate the superiority of the proposed method over existing methods. In the future, we plan to extend the proposed idea to other related tasks.

References

Almazán, J.; Gordo, A.; Fornés, A.; and Valveny, E. 2014. Word Spotting and Recognition with Embedded Attributes. *IEEE TPAMI* 36(12):2552–2566.

Alsharif, O., and Pineau, J. 2014. End-to-end text recognition with hybrid hmm maxout models. In *ICLR*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*.

Bissacco, A.; Cummins, M.; Netzer, Y.; and Neven, H. 2013. PhotoOCR: Reading Text in Uncontrolled Conditions. In *ICCV*, 785–792.

Bookstein, F. L. 1989. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE TPAMI* 11(6):567–585.

Cheng, Z.; Bai, F.; Xu, Y.; Zheng, G.; Pu, S.; and Zhou, S. 2017. Focusing Attention: Towards Accurate Text Recognition in Natural Images. *arXiv preprint arXiv:1709.02054*.

Chorowski, J. K.; Bahdanau, D.; Serdyuk, D.; Cho, K.; and Bengio, Y. 2015. Attention-Based Models for Speech Recognition. In *NIPS*, 577–585.

Goel, V.; Mishra, A.; Alahari, K.; and Jawahar, C. V. 2013. Whole is Greater than Sum of Parts: Recognizing Scene Text Words. In *ICDAR*, 398–402.

Gordo, A. 2015. Supervised mid-level features for word image representation. In *CVPR*, 2956–2964.

Graves, A.; Fernández, S.; Gomez, F.; and Schmidhuber, J. 2006. Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *ICML*, 369–376. ACM.

Graves, A.; r. Mohamed, A.; and Hinton, G. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*, 6645–6649.

Gupta, A.; Vedaldi, A.; and Zisserman, A. 2016. Synthetic Data for Text Localisation in Natural Images. In *CVPR*, 2315–2324.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

He, P.; Huang, W.; Qiao, Y.; Loy, C. C.; and Tang, X. 2016b. Reading Scene Text in Deep Convolutional Sequences. In *AAAI*, 3501–3508.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 448–456.

Jaderberg, M.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. *arXiv preprint arXiv:1406.2227*.

Jaderberg, M.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2015a. Deep Structured Output Learning for Unconstrained Text Recognition. In *ICLR*.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; and Kavukcuoglu, K. 2015b. Spatial Transformer Networks. *NIPS* 2017–2025.

Jaderberg, M.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2016. Reading Text in the Wild with Convolutional Neural Networks. *IJCV* 116(1):1–20.

Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; and Darrell, T. 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In *ACM-MM*, 675–678.

Jiang, Y.; Zhu, X.; Wang, X.; Yang, S.; Li, W.; Wang, H.; Fu, P.; and Luo, Z. 2017. R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. *arXiv preprint arXiv:1706.09579*.

Karatzas, D.; Gomez-Bigorda, L.; Nicolaou, A.; Ghosh, S.; Bagdanov, A.; Iwamura, M.; Matas, J.; Neumann, L.; Chandrasekhar, V. R.; Lu, S.; Shafait, F.; Uchida, S.; and Valveny, E. 2015. ICDAR 2015 competition on Robust Reading. In *ICDAR*, 1156–1160.

Lee, C. Y., and Osindero, S. 2016. Recursive Recurrent Nets with Attention Modeling for OCR in the Wild. In *CVPR*, 2231–2239.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*, 3431–3440.

Lucas, S. M.; Panaretos, A.; Sosa, L.; Tang, A.; Wong, S.; and Young, R. 2003. ICDAR 2003 robust reading competitions. In *ICDAR*, 682–687.

Ma, J.; Shao, W.; Ye, H.; Wang, L.; Wang, H.; Zheng, Y.; and Xue, X. 2017. Arbitrary-Oriented Scene Text Detection via Rotation Proposals. *arXiv preprint arXiv:1703.01086*.

Mishra, A.; Alahari, K.; and Jawahar, C. V. 2012. Scene Text Recognition using Higher Order Language Priors. In *BMVC*, 1–11.

Neumann, L., and Matas, J. 2012. Real-time scene text localization and recognition. In *CVPR*, 3538–3545.

Quy Phan, T.; Shivakumara, P.; Tian, S.; and Lim Tan, C. 2013. Recognizing text with perspective distortion in natural scenes. In *ICCV*, 569–576.

Risnumawan, A.; Shivakumara, P.; Chan, C. S.; and Tan, C. L. 2014. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications* 41(18):8027–8048.

Shi, B.; Bai, X.; and Belongie, S. 2017. Detecting Oriented Text in Natural Images by Linking Segments. *arXiv preprint arXiv:1703.06520*.

Shi, B.; Bai, X.; and Yao, C. 2016. An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE TPAMI* preprint.

Shi, B.; Wang, X.; Lyu, P.; Yao, C.; and Bai, X. 2016. Robust Scene Text Recognition with Automatic Rectification. In *CVPR*, 4168–4176.

Su, B., and Lu, S. 2015. Accurate Scene Text Recognition Based on Recurrent Neural Network. In *ACCV*, 35–48.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to Sequence Learning with Neural Networks. In *NIPS*, 3104–3112.

Wang, K., and Belongie, S. 2010. Word Spotting in the Wild. In *ECCV*, 591–604. Springer.

Wang, K.; Babenko, B.; and Belongie, S. 2011. End-to-end scene text recognition. In *ICCV*, 1457–1464.

Wang, T.; Wu, D. J.; Coates, A.; and Ng, A. Y. 2012. End-to-end text recognition with convolutional neural networks. In *ICPR*, 3304–3308.

Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*, 2048–2057.

Yang, X.; He, D.; Zhou, Z.; Kifer, D.; and Giles, C. L. 2017. Learning to Read Irregular Text with Attention Mechanisms. In *IJCAI*, 3280–3286.

Yao, C.; Bai, X.; Shi, B.; and Liu, W. 2014. Strokelets: A Learned Multi-scale Representation for Scene Text Recognition. In *CVPR*, 4042–4049.

Ye, Q., and Doermann, D. 2015. Text Detection and Recognition in Imagery: A Survey. *IEEE TPAMI* 37(7):1480–1500.

Zeiler, M. D. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR* abs/1212.5701.

Zhang, Z.; Zhang, C.; Shen, W.; Yao, C.; Liu, W.; and Bai, X. 2016. Multi-oriented text detection with fully convolutional networks. In *CVPR*, 4159–4167.