Focusing Attention: Towards Accurate Text Recognition in Natural Images

Zhanzhan Cheng[1] Fan Bai[2] Yunlu Xu[3] Gang Zheng[1]
Shiliang Pu[1] Shuigeng Zhou[2*]
[1]Hikvision Research Institute, China; [2]Shanghai Key Lab of Intelligent Information Processing and School of Computer Science, Fudan University, Shanghai, China;
[3]Shanghai Jiaotong University, Shanghai, China
{chengzhanzhan;zhenggang3;pushiliang}@hikvision.com;
{baif13;sgzhou}@fudan.edu.cn; xuyunlu1030@163.com

Abstract

*Scene text recognition has been a hot research topic in computer vision due to its various applications. The state of the art is the attention-based encoder-decoder framework that learns the mapping between input images and output sequences in a purely data-driven way. However, we observe that existing attention-based methods perform poorly on complicated and/or low-quality images. One major reason is that existing methods cannot get accurate alignments between feature areas and targets for such images. We call this phenomenon "attention drift". To tackle this problem, in this paper we propose the **FAN** (the abbreviation of **F**ocusing **A**ttention **N**etwork) method that employs a focusing attention mechanism to automatically draw back the drifted attention. FAN consists of two major components: an attention network (AN) that is responsible for recognizing character targets as in the existing methods, and a focusing network (FN) that is responsible for adjusting attention by evaluating whether AN pays attention properly on the target areas in the images. Furthermore, different from the existing methods, we adopt a ResNet-based network to enrich deep representations of scene text images. Extensive experiments on various benchmarks, including the IIIT5k, SVT and ICDAR datasets, show that the FAN method substantially outperforms the existing methods.*

1. Introduction

Scene text recognition has attracted much research interest of the computer vision community [11, 21, 24]. Recognizing scene text is of great significance for scene understanding. Despite several decades of research on Optical Character Recognition (OCR), recognizing texts from natural images is still a challenging task. The state of the art employs attention-mechanism for character recognition, and achieves substantial performance improvement [18, 25].

Usually, an attention-based text recognizer is designed as an encoder-decoder framework. In the encoding stage, an image is transformed into a sequence of feature vectors by CNN/LSTM [25], and each feature vector corresponds to a region in the input image. In this paper, we call such regions *attention regions*. In the decoding stage, the attention network (AN) first computes alignment factors [3] by referring to the history of target characters and the encoded feature vectors for generating the synthesis vectors (also called glimpse vectors), thus achieves the alignments between the attention regions and the corresponding ground-truth-labels [3, 5]. Then, a recurrent neural network (RNN) is used to generate the target characters based on the glimpse vectors and the history of target characters.

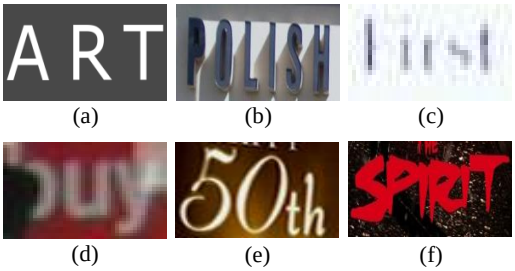

Figure 1. Examples of complicated / low-quality images. Subfigures (a) - (f) represent normal, complex background, blur, incomplete, different-size and abnormal font images, respectively.

Motivation. We all know that in real scene text recognition tasks, many images are complicated (*e.g.* distorted or overlapping characters, characters of different fonts, sizes and colors, and complex backgrounds) or low quality (due

*Corresponding author.

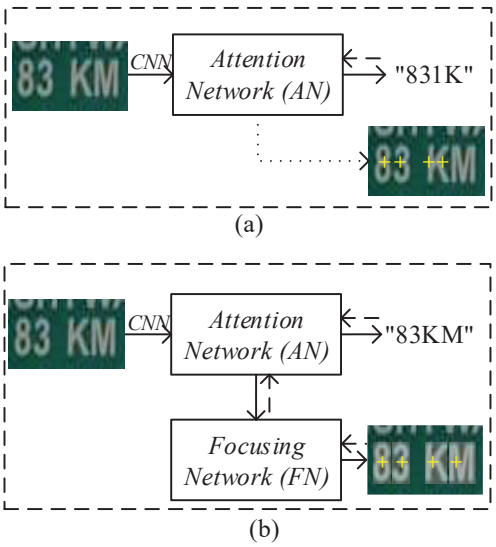

Figure 2. Illustration of attention drift in the AN model and the focusing mechanism in the FAN method. In sub-figure (a), a real image with text "83KM" is taken as input, and it outputs "831K". The last two characters 'K' and 'M' are not recognized because AN's attention regions for these two characters are deviated a lot from them in the image. In sub-figure (b), with the FN component, AN's attention centers for the last two characters are rectified and positioned just on them, thus FAN outputs the correct text string "83KM". Here, the dotted arrows and yellow '+'s represent the computation of attention regions and the centers of attention regions, and the white rectangular masks in the right-bottom image indicate the ground-truth-areas of the characters.

to illumination change, blur, incompleteness and noise etc.) Fig. 1 shows some examples of complicated / low-quality images. For such images, existing attention-based methods often perform poorly. After carefully analyzing many intermediate and final results of attention-based methods on real data, we found that one major reason of poor performance is because the alignments estimated by the attention model are easily corrupted due to the complexity and/or low-quality of images. In other words, the attention model cannot accurately associate each feature vector with the corresponding target region in the input image. We call this phenomenon *attention drift*. That is, the attention regions of AN deviate in some degree from the proper regions of target characters in the images. This motivates us to develop some mechanism to focus AN's attention back on the right regions of the target characters in the input image.

Fig. 2(a) illustrates the attention drift phenomenon in the AN model. After the left image is input, we expect that the AN model outputs a text string "83KM", but actually it returns "831K". Note that this is not a toy example, it is a real example selected from our experiments. In practice, there are many of such examples. Obviously, the last two characters 'K' and 'M' are not correctly recognized. How this happens? By computing the attention regions of the four characters in the image, we can get their attention centers (the details are given in Section 3.2), which are illustrated by yellow '+' in the right-bottom corner with the original image as background. We can see that the attention centers of '8' and '3' are located just over themselves, while the third attention center is on the left-half part of 'K' and the fourth attention center is near the right half part of 'K'. As the left-half part of 'K' looks like a '1', the AN model outputs a '1'. The fourth attention region covers most part of 'K', the AN model thus returns a 'K'.

Our work. To solve the above problem, in this paper we propose a novel method called FAN (the abbreviation of Focusing Attention Network) to accurately recognize text from natural images. Fig. 2(b) shows the architecture of the FAN method. FAN is made of two major subnetworks: an attention network (AN) that is to recognize the target characters as in existing methods, and a *focusing network* (FN) that is responsible for automatically adjusting the attention of AN by first examining whether AN's attention regions are properly on the right regions of target characters in the images. In Fig. 2(b), with the FN component, AN's attention regions for the last two characters are rectified, and consequently FAN outputs the correct text string "83KM".

Contributions of this paper are as follows:

1) We propose the concept of attention drift, which explains the poor performance of existing attention based methods on complicated and/or low-quality natural images.

2) We develop a novel method called FAN to solve the attention drift problem, where in addition to the AN component existing in most existing methods, a completely new component — focusing network (FN) is introduced, which can focus AN's deviated attention back on the target areas.

3) We adopt a powerful ResNet-based [10] convolution neural network (CNN) to enrich deep representations of scene text images.

4) We conduct extensive experiments on several benchmarks, which demonstrate the performance superiority of our method over the existing methods.

2. Related work

In recent years, there has been a lot work on scene text recognition. For general information of text recognition, the readers can refer to Ye and Doermann's recent survey [33]. Traditionally, there are two types of approaches: bottom-up and top-down. Early works mainly develop *bottom-up* approaches: first detecting characters one by one by sliding window [29, 30], connected components [21] or Hough voting [32], then integrating these characters into the output text. 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*. designed a convolutional neural network (CNN) with structured output layer for unconstrained recognition [12]. They also conducted a 90k-class classification task with a CNN, in which each class represents an English word [13] . 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 *et al*. [28] extracted sequences of HOG features to represent images, and generated the character sequence with the recurrent neural network (RNN). Shi *et al*. [24] proposed an end-to-end neural network that combines CNN and RNN. They also developed an attention-based spatial transformer network (STN) for rectifying text distortion, which helps recognize curved scene texts [25].

Different from the existing approaches, in this paper we first extract deeper representations of images by using a ResNet-based CNN. To the best of our knowledge, this may be the first work of scene text recognition that uses a ResNet-based CNN. Then, we feed the sequence of features to an AN for generating alignment factors and *glimpse vectors*. Meanwhile, we employ a FN to evaluate whether the *glimpse vectors* are reasonable and provide a feedback to help AN generate more reasonable *glimpse vectors* so that the AN can pay attention properly on the right regions of target characters in the processed image.

Though *attention drift* has been observed in attention training of speech recognition [17], where the authors proposed an MTL framework that combines CTC and AN to handle this issue, our paper is the first work that formally puts forward the concept of *attention drift*. Furthermore, we design a focus-mechanism to solve this problem. It is worth of noting that we have tried to use CTC and AN to solve the attention drift problem in scene text recognition, unfortunately our extensive experiments showed that this idea does not work well, so we discarded it.

3. The FAN Method

As shown in Fig. 2(b), FAN has two major modules: AN and FN. In the AN component, alignment factors [3] between target labels and features are generated. Each alignment factor corresponds to an attention region in the input image. Bad alignments (*i.e*. deviated or unfocused attention regions) result in poor recognition results. The FN component first locates the attention region for each target label, then conducts dense prediction over the attention region with the corresponding *glimpse vector*. In such a way, FN can judge whether the glimpse vector is reasonable. In summary, FN generates the dense outputs over the attention regions in the input image based on the *glimpse vectors* provided by AN, and AN in turn updates the *glimpse vectors* based on FN's feedback.

3.1. Attention Network (AN)

An attention-based decoder is a recurrent neural network (RNN) that directly generates the target sequence $(y_1, ..., y_M)$ from an input image $\mathcal{I}$. In practice, image $\mathcal{I}$ is often encoded as a sequence of feature vectors by CNN-LSTM. Formally, $Encoder(\mathcal{I})$=$(h_1, ..., h_T)$. Bahdanau *et al*. [3] first proposed the architecture of attention-based decoder. At the t-th step, the decoder generates an output y_t

$$y_t = Generate(s_t, g_t), \tag{1}$$

where s_t is an RNN hidden state at time t, computed by

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

and g_t is the weighted sum of sequential feature vectors $(h_1, ..., h_T)$, that is,

$$g_t = \sum_{j=1}^{T} \alpha_{t,j} h_j, \tag{3}$$

where $\alpha_t \in \mathbb{R}^T$ is a vector of *attention weights*, also called *alignment factors*. α_t is often evaluated by scoring each element in $(h_1, ..., h_T)$ separately and then normalizing the scores as follows:

$$e_{t,j} = v^T \tanh(W s_{t-1} + V h_j + b), \tag{4}$$

$$\alpha_{t,j} = \frac{exp(e_{t,j})}{\sum_{j=1}^{T} exp(e_{t,j})}. \tag{5}$$

Above, v, W, V and b are all trainable parameters.

Here, the $Generate$ function in Eq. (1) and the RNN function in Eq. (2) represent a feed-forward network and a LSTM recurrent network, respectively. Besides, the decoder needs to generate sequences of variable lengths. Following [28], a special end-of-sentence (EOS) token is added to the target set, so that the decoder completes the generation of characters when EOS is emitted. The loss function of the attention model is as follows:

$$\mathcal{L}_{Att} = -\sum_{t} lnP(\hat{y}_t | \mathcal{I}, \theta), \tag{6}$$

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

In scene text recognition, the AN model has two major drawbacks: 1) This model is easily impacted by complicated / low-quality scene data, and generates imprecise alignment factors because the model has no alignment constraint on the integration of *glimpse vectors*, which may result in mismatch between attention regions and ground-truth regions. This is the so-called attention drift problem mentioned above. 2) It is hard to train such a model on huge

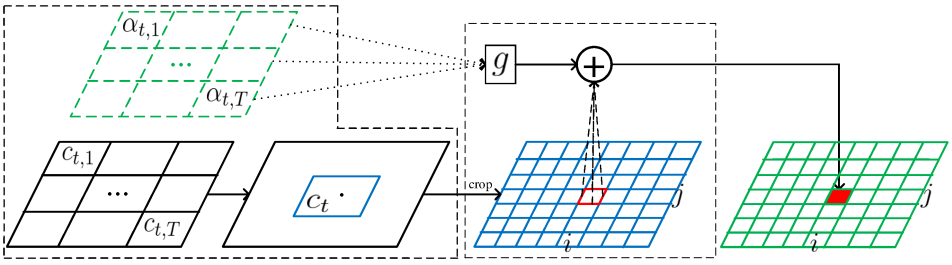

Figure 3. The mechanism of attention focusing in FAN. Here, α, c, g and $+$ represent alignment factors, center of each feature in the input image, *glimpse vector* and focusing operation, respectively. The blue grid and the green grid indicate the cropped features and predicted results over each pixel, respectively. In order to predict the t-*th* target, we first evaluate the center position $c_{t,j}$ for each feature vector h_j obtained by *CNN-LSTM*, and compute weighted sum of all centers to get a weighted position c_t, then crop a patch of features from the input image or a convolution output and do the focusing operation over the attention region.

scene text data, such as the 800-million synthetic data released by Gupta *et al*. [9]. In this paper, our major goal is to tackle the attention drift problem. We try to constrain AN's attention just on each target character by introducing the focusing network, which is detailed in the following section.

3.2. Focusing Network (FN)

In the attention model, each feature vector is mapped to an area of the input image, which can be used to localize the target character based on the convolution strategy. However, the computed attention regions of targets are usual inaccurate, especially for complicated and/or low-quality images. In order to handle the *attention drift* problem, we introduce a focusing network with a focusing-mechanism to rectify the drifted attention. The focusing-mechanism is illustrated in Fig. 3. It works in two major steps: 1) computing the attention center of each predicted label; 2) focusing attention on target regions by generating the probability distributions on the attention regions.

Computing attention center: In convolution/pooling operation, we define the input as $N \times D_i \times H_i \times W_i$, and the output as $N \times D_o \times H_o \times W_o$, where N, D, H and W indicate the batch size, the number of channels, the height and width of feature maps, respectively. With the convolution strategy: $kernel$, $stride$ and pad, we have $H_o = (H_i + 2 \times pad_H - kernel_H)/stride_H + 1$ and $W_o = (W_i + 2 \times pad_W - kernel_W)/stride_W + 1$. Therefore, for position (x, y) in layer L, we compute its receptive field in layer $L-1$ as the bounding box coordinates $r = (x_{min}, x_{max}, y_{min}, y_{max})$ as follows:

$$\begin{aligned} x_{min} &= (x-1) \times stride_W + 1 - pad_W, \\ x_{max} &= (x-1) \times stride_W - pad_W + kernel_W, \\ y_{min} &= (y-1) \times stride_H + 1 - pad_H, \\ y_{max} &= (y-1) \times stride_H - pad_H + kernel_H. \end{aligned} \tag{7}$$

At the t-th step, we compute the receptive field of h_j in the input image by recursively performing the computation of Eq. (7), and select the center of the receptive field as the attention center:

$$c_{t,j} = location(j) \tag{8}$$

where j refers the index of h_j, and $location$ denotes the function of evaluating the center of a receptive field. Therefore, the attention center of target y_t in the input image is evaluated as follows:

$$c_t = \sum_{j=1}^{T} \alpha_{t,j} c_{t,j}. \tag{9}$$

Focusing attention on target regions: With the computed attention center of target y_t, we crop a patch of feature maps of size $\mathcal{P}(\mathcal{P}_H, \mathcal{P}_W)$ from the input image or a convolution output as follows:

$$\mathcal{F}_t = Crop(\mathcal{F}, c_t, \mathcal{P}_H, \mathcal{P}_W) \tag{10}$$

where $\mathcal{F}$ is the image/convolution feature maps, $\mathcal{P}$ is set to the max-size of ground-truth regions in the input image.

With the cropped feature maps, we compute the energy distribution over the attention region as follows:

$$e_t^{(i,j)} = \tanh(Rg_t + S\mathcal{F}_t^{(i,j)} + b) \tag{11}$$

Above, R and S are trainable parameters, and (i, j) refers to the $(i \times \mathcal{P}_W + j)$-th feature vector. Then, the probability distribution over the selected region is computed as

$$P_t^{(i,j,k)} = \frac{exp(e_t^{(i,j,k)})}{\sum_{k'}^{K} exp(e_t^{(i,j,k')})}, \tag{12}$$

where K is the number of label classes.

Then, we define the *focusing loss function* as

$$\mathcal{L}_{focus} = -\sum_{t}^{M} \sum_{i}^{\mathcal{P}_W} \sum_{j}^{\mathcal{P}_H} logP(\hat{y}_t^{(i,j)} | \mathcal{I}, \omega) \tag{13}$$

where $\hat{y}_t^{(i,j)}$ is the ground-truth pixel label and ω is a vector that combines all the FN parameters. The loss is added only for the subset of images with character annotations.

3.3. FAN Training

We combine a ResNet-based feature extractor, AN and FN into one network, as shown in Fig. 4. The details are given in Section 4.2. AN uses the extracted features to generate alignment factors and glimpse vectors, with which FN focuses the attention of AN on the proper target character regions in the images.

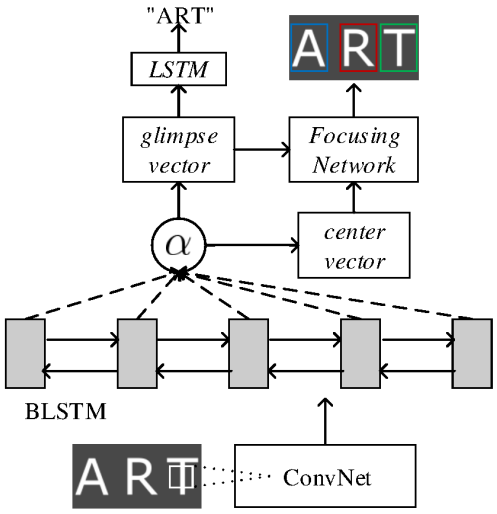

Figure 4. The FAN network architecture. Here, the *CNN-BLSTM* encoder transforms an input image $\mathcal{I}$ into high level sequence of features, the RNN decoder generates each target character, and FN focuses the attention of AN on the right target character regions in the input images. FN and AN are trained simultaneously.

The objective function is constructed by considering both target-generation and attention-focusing as follows:

$$\mathcal{L} = (1-\lambda)\mathcal{L}_{Att} + \lambda\mathcal{L}_{focus}, \tag{14}$$

with a tunable parameter λ ($0 \leqslant \lambda < 1$), which trades off the impact of AN and FN. The network is trained by standard back-propagation [23].

3.4. Decoding

The attention-based decoder is to generate the output sequence of characters from the implicitly learned character-level probability statistics. In the process of unconstrained text recognition (lexicon-free), we straightforwardly select the most probable character. While in constrained text recognition, according to the lexicons of different sizes (denoted by "50", "1k" and "full"), we calculate the conditional probability distributions for all lexicon words, and take the one with the highest probability as output result.

layer name	32 layers	output size
conv1_x	$3\times3, 1\times1, 1\times1, 32$	32×256
	$3\times3, 1\times1, 1\times1, 64$	
conv2_x	pool:$2\times2, 2\times2, 0\times0$	16×128
	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times1$	
	$3\times3, 1\times1, 1\times1, 128$	
conv3_x	pool:$2\times2, 2\times2, 0\times0$	8×64
	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times2$	
	$3\times3, 1\times1, 1\times1, 256$	
conv4_x	pool:$2\times2, 1\times2, 1\times0$	4×65
	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times5$	
	$3\times3, 1\times1, 1\times1, 512$	
conv5_x	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times3$	1×65
	$2\times2, 1\times2, 1\times0, 512$	
	$2\times2, 1\times1, 0\times0, 512$	

Table 1. A ResNet-based CNN architecture for robust text feature extraction. Building blocks are shown in brackets, and ResNet blocks are highlighted with gray background.

4. Performance Evaluation

We conduct extensive experiments to validate the proposed FAN method on a number of general recognition benchmarks commonly used in the literature. For comprehensive performance comparison, FAN is compared with 18 existing methods and the AN model implemented with a ResNet-based encoder, which is taken as the baseline method. To further verify the effectiveness of our attention focusing mechanism, we compare FAN with AN in the context that both are based on the image-encoder released by Shi *et al*. [25]. Finally, we also investigate the effect of two major parameters on the performance of FAN.

4.1. Datasets

IIIT 5K-Words [20] (IIIT5K) 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 [29] (SVT) 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.

Method	IIIT5k			SVT		IC03			IC13	IC15
	50	1k	None	50	None	50	Full	None	None	None
ABBYY [29]	24.3	–	–	35.0	–	56.0	55.0	–	–	–
Wang *et al*. [29]	–	–	–	57.0	–	76.0	62.0	–	–	–
Mishra *et al*. [8]	64.1	57.5	–	73.2	–	81.8	67.8	–	–	–
Wang *et al*. [31]	–	–	–	70.0	–	90.0	84.0	–	–	–
Goel *et al*. [6]	–	–	–	77.3	–	89.7	–	–	–	–
Bissacco *et al*. [4]	–	–	–	90.4	78.0	–	–	–	87.6	–
Alsharif and Pineau [2]	–	–	–	74.3	–	93.1	88.6	–	–	–
Almazán *et al*. [1]	91.2	82.1	–	89.2	–	–	–	–	–	–
Yao *et al*. [32]	80.2	69.3	–	75.9	–	88.5	80.3	–	–	–
Rodríguez-Serrano *et al*. [22]	76.1	57.4	–	70.0	–	–	–	–	–	–
Jaderberg *et al*. [12]	–	–	–	86.1	–	96.2	91.5	–	–	–
Su and Lu [26]	–	–	–	83.0	–	92.0	82.0	–	–	–
Gordo [7]	93.3	86.6	–	91.8	–	–	–	–	–	–
Jaderberg *et al*. [13]	97.1	92.7	–	95.4	80.7	98.7	**98.6**	93.1	90.8	–
Jaderberg *et al*. [12]	95.5	89.6	–	93.2	71.7	97.8	97.0	89.6	81.8	–
Shi *et al*. [24]	97.6	94.4	78.2	96.4	80.8	98.7	97.6	89.4	86.7	–
Shi *et al*. [25]	96.2	93.8	81.9	95.5	81.9	98.3	96.2	90.1	88.6	–
Baidu IDL. [27]	–	–	–	–	–	–	–	–	89.9	**77.0**
Baseline	98.9	96.8	83.7	95.7	82.2	98.5	96.7	91.5	89.4	63.3
FAN	**99.3**	**97.5**	**87.4**	**97.1**	**85.9**	**99.2**	97.3	**94.2**	**93.3**	70.6

Table 2. Recognition accuracies on general benchmarks. "50" and "1k" are lexicon sizes, "Full" indicates the combined lexicon of all images in the benchmarks, and "None" means lexicon-free.

ICDAR 2003 [19] (IC03) contains 251 scene images, labeled with text bounding boxes. Each image is associated with a 50-word lexicon defined by Wang *et al*. [29]. For fair comparison, we discard images that contain non-alphanumeric characters or have less than three characters, following [29]. The resulting dataset contains 867 cropped images. The lexicons include the 50-word lexicons and the full lexicon which combines all lexicon words.

ICDAR 2013 [16] (IC13) is the successor of IC03, from which most of its data are inherited. It contains 1015 cropped text images. No lexicon is associated.

ICDAR 2015 [15] (IC15) contains 2077 cropped images. For fair comparison, we discard the images that contain non-alphanumeric characters, which results in 1811 images. No lexicon is associated.

4.2. Implementation Details

Network: For the encoder, we construct a 32-layer ResNet [10]-based CNN as described in Tab. 1 for extracting deeper represented text features. All the ResNet blocks with gray background in Tab. 1 have the following format: $\{[kernel\ size, number\ of\ channels]\times\}$, each of which has $\{stride, pad\} = \{0, 0\}$. The other convolution layers have the following format: $\{kernel_W \times kernel_H, stride_W \times stride_H, pad_W \times pad_H, channels\}$; And the maxpooling layers have the following format: $pool : \{kernel_W \times kerne_H, stride_W \times stride_H, pad_W \times pad_H\}$. The subscript H and W respectively represent the height and width of feature maps. Then, the extracted sequence of features from CNN are fed into a BLSTM (256 hidden units) network. For the character generation task, the attention is designed with a LSTM (256 memory blocks) and 37 output units (26 letters, 10 digits, and 1 EOS symbol). For FAN, we crop feature maps from the input image and set $\lambda = 0.01$. We will further discuss the tuning of parameter λ in Section 4.4.

Model Training: With the ADADELTA [34] optimization method, we train our model on 8-million synthetic data without being pixel-wise labeled released by Jaderberg *et al*. [11] and 4-million synthetic pixel-wise labeled word instances (excluding the images that contain non-alphanumeric characters) cropped from 80-thousand images released by Gupta *et al*. [9]. That is, about 30% instances have pixel labels. We set the batch size to 32 and scale images to 256×32 in both training and testing. Our model processes about 90 samples per second, and converges in 5 days after about 3 epochs over the training set.

Implementation and Running Environment Our method is implemented under the CAFFE framework [14]. We use the CUDA backend extensively in our implementation, so that most modules in our model are GPU-accelerated. Our experiments are carried out on a workstation with one Intel Xeon(R) E5-2650 2.30GHz CPU, an NVIDIA Tesla M40 GPU, and 128GB RAM.

Method	IIIT5k			SVT		IC03			IC13	IC15
	50	1k	None	50	None	50	Full	None	None	None
Baseline	31.7	78.0	188.9	22.9	47.3	11.2	18.4	28.5	58.1	306.8
FAN	**22.0**	**64.5**	**145.2**	**16.3**	**36.7**	**6.0**	**16.2**	**21.3**	**39.8**	**251.9**

Table 3. The total normalized edit distance results on general benchmarks. "50" and "1k" are lexicon sizes, "Full" indicates the combined lexicon of all images in the dataset, and "None" means lexicon-free.

4.3. Performance on General Recognition Datasets

Tab. 2 shows the performance results of our method and 18 existing methods as well as a *baseline method* that combines an AN and a ResNet-based feature extractor. The only difference between the FAN method and the baseline is that FAN has an FN component. So comparing the results of the baseline and FAN, we can infer the performance contribution of the FN component to the FAN method.

From Tab. 2, we can see that FAN achieves better performance than the baseline in all cases, and substantially outperforms the 18 existing methods on almost all benchmarks, but IC03 with 'Full' lexicon and IC15. This demonstrates the effectiveness and superiority of the FAN method. Moreover, for both constrained and unconstrained cases, the baseline method significantly outperforms all the existing methods on IIIT5k, SVT, and performs comparably to the best existing method on IC03 and IC13, but falls behind [27] released on the ICDAR15 competition. In fact, our model don't conduct fine-tune on the IC15 train set.Thus we can conclude that the ResNet-based network indeed extracts more robust features for scene text recognition and greatly boosts the recognition performance.

In the ICDAR competition, the sum of normalized edit distance (NED) [15] of each image is used as the final indicator for method ranking. We also evaluate the total normalized edit distance (NED) on all benchmarks for the baseline and FAN, and the results are shown in Tab. 3. Here, NED is simply defined as $edit\_distance(pred, gt)/|gt|$, where $pred$ and gt represent the prediction result and ground truth respectively. Comparing to the baseline method, we can see that FAN significantly improves the NED measure in both constrained and unconstrained cases.

Fig. 5 shows some real images processed by AN and FAN. Comparing with the outputs of AN, FAN obviously rectify the attention drift problem and correctly recognize more characters in the images.

Method	IIIT5k	SVT	IC03	IC13	IC15
AN	248.0	65.9	43.1	85.2	348.2
FAN	**213.4**	**56.0**	**31.5**	**72.1**	**323.6**

Table 4. Total NED results of AN and FAN on unconstrained benchmarks with the image-encoder released by Shi *et al*. [25].

As we know, FAN consists of three major components:

Figure 5. Some real images processed by AN and FAN. The left and right columns are the output results of AN and FAN respectively. The predicted text string for each input image is shown just below the image. The attention centers are marked as '+' of yellow (for AN) and green (for FAN). Green/red characters indicate correctly/incorrectly recognized characters.

Method	IIIT5k	SVT	IC03	IC13	IC15
AN	79.5	75.6	88.2	85.8	58.2
FAN	**81.7**	**78.7**	**90.3**	**87.8**	**61.0**

Table 5. Accuracy results of AN and FAN on unconstrained benchmarks with the image-encoder released by Shi *et al*. [25].

AN, FN and a ResNet-based encoder. To further evaluate the performance contribution of FN to FAN, here we compare the performance of FAN and AN, which are all based

on the image-encoder released by Shi *et al*. [25], instead of the ResNet-based encoder, over the five unconstrained datasets (IIIT5K, SVT, IC03, IC13 and IC15). That is, both FAN and AN use the image-encoder released by Shi *et al*. [25] to do text recognition. The results are presented in Tab. 4 and Tab 5, which show that FAN still significantly outperforms AN in terms of accuracy and total NED. This demonstrates again the outstanding performance of our attention focusing mechanism even when the ResNet-based encoder is not used.

4.4. The Effect of Parameter λ

We need to tune the super-parameter λ in our training objective function Eq. (14) for better performance of the network. λ trades off the impact of AN and FN. Here, we explore how λ impacts FAN's performance.

λ	IIIT5k	SVT	IC03	IC13	IC15
0.001	153.2	40.5	23.3	44.6	262.9
0.005	147.5	37.6	22.8	42.2	255.3
0.01	**145.2**	**36.7**	**21.3**	**39.8**	**251.9**
0.05	148.4	38.1	23.1	45.2	252.5
0.1	150.0	38.4	23.3	49.8	253.8

Table 6. The total NED results on unconstrained benchmarks with different λ values.

λ	IIIT5k	SVT	IC03	IC13	IC15
0.001	86.6	85.8	93.2	92.7	69.5
0.005	87.2	85.6	93.5	92.8	70.2
0.01	**87.4**	85.9	**94.2**	**93.3**	70.6
0.05	87.0	**86.2**	94.1	92.5	70.6
0.1	86.9	86.2	93.9	92.1	**70.7**

Table 7. Accuracy results on unconstrained benchmarks with different λ values.

Intuitively, λ should be from 0 to 1. A larger value means the FN component plays a more important role in the text recognition task, and vice versa. $\lambda = 0$ means that FN does not work in the recognition process, *i.e*., the focusing mechanism is not employed. On the contrary, $\lambda = 1$ means that AN does not work, which is unacceptable. In our experiments, we vary λ from 0 to 0.1, and the results on the unconstrained recognition benchmarks are given in Tab. 6 and Tab. 7. We can see that FAN performs stably and achieves relatively higher performance with $\lambda = 0.01$.

4.5. The Effect of Pixel Labeling

Pixel labeling for characters in the training images certainly benefits recognition performance, but also consumes large amount of resource (time and money). Therefore, here we examine how the amount of pixel labeling impacts the recognition performance. We expect that FAN trained with a small number of pixel-labeled (or positioned) samples can still achieve a high performance.

Ratio	IIIT5k	SVT	IC03	IC13	IC15
0%	188.9	47.3	28.5	58.1	306.8
1%	155.3	41.7	22.6	46.3	258.3
5%	152.5	41.8	22.3	52.7	258.3
10%	152.9	39.7	22.1	43.5	264.9
30%	**145.2**	**36.7**	**21.3**	**39.8**	**251.9**

Table 8. The total NED results on unconstrained benchmarks for different numbers of positioned-samples.

Ratio	IIIT5k	SVT	IC03	IC13	IC15
0%	83.7	82.2	91.5	89.4	63.3
1%	86.2	84.4	93.8	92.7	70.0
5%	86.3	84.7	93.8	92.6	69.4
10%	86.3	84.5	94.0	93.1	70.0
30%	**87.4**	**85.9**	**94.2**	**93.3**	**70.6**

Table 9. Accuracy results on unconstrained benchmarks for different numbers of positioned-samples.

Tab. 8 and Tab. 9 show the results of NED and accuracy on unconstrained benchmarks by increasing the ratio of pixel-labeled samples from 0 to 30%. We can see that even with only a small fraction of pixel-labeled samples in the training set, our method can still guide the model to achieve satisfactory performance.

5. Conclusion

In this work, we put forward the *attention drift* concept to explain the poor performance of existing AN based methods of scene text recognition on complicated and/or low-quality images, and propose a novel method FAN to solve this problem. Different from the existing methods, FAN uses an innovative focusing network to rectify the drifted attention of the AN model in handling complicated and low-quality images. Extensive experiments over several benchmarks show that the proposed method significantly outperforms existing methods. In the future, we plan to extend the proposed idea to text detection and other related tasks.

Acknowledgement

Fan Bai and Shuigeng Zhou were partially supported by the Key Projects of Fundamental Research Program of Shanghai Municipal Commission of Science and Technology under grant No. 14JC1400300.

References

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

[16] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i. Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazàn, and L. P. de las Heras. ICDAR 2013 Robust Reading Competition. In *ICDAR*, pages 1484–1493, 2013.

[17] S. Kim, T. Hori, and S. Watanabe. Joint CTC-Attention based End-to-End Speech Recognition using Multi-task Learning. *arXiv preprint arXiv:1609.06773*, 2016.

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

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

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

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

[22] J. A. Rodriguez-Serrano, A. Gordo, and F. Perronnin. Label Embedding: A Frugal Baseline for Text Recognition. *IJCV*, 113(3):193–207, 2015.

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive Modeling*, 5(3):1, 1988.

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

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

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

[27] Z. Su, H. Xiao, W. Liu, S. Xie, J. Han, and D. Errui. Joint trained CNN and attention based RNN are applied to recognize the words. `http://rrc.cvc.uab.es/?ch=4&com=evaluation`. Accessed: 2017-3-17.

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

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

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

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

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

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

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