

Boosting Scene Character Recognition by Learning Canonical Forms of Glyphs

Yizhi Wang · Zhouhui Lian · Yingmin Tang · Jianguo Xiao

Received: date / Accepted: date

Abstract As one of the fundamental problems in document analysis, scene character recognition has attracted considerable interests in recent years. But the problem is still considered to be extremely challenging due to many uncontrollable factors including glyph transformation, blur, noisy background, uneven illumination, etc. In this paper, we propose a novel methodology for boosting scene character recognition by learning canonical forms of glyphs, based on the fact that characters appearing in scene images are all derived from their corresponding canonical forms. Our key observation is that more discriminative features can be learned by solving specially-designed generative tasks compared to traditional classification-based feature learning frameworks. Specifically, we design a GAN-based model to make the learned deep feature of a given scene character be capable of reconstructing corresponding glyphs in a number of standard font styles. In this manner, we obtain deep features for scene characters that are more discriminative in recognition and less sensitive against the above-mentioned factors. Our experiments conducted on several publicly-available databases demonstrate the superiority of our method compared to the state of the art.

Keywords Character recognition · Canonical forms of glyphs · Multi-task learning · Deep learning

1 Introduction

Scene character recognition (SCR) is an important and challenging problem in areas of Document Analysis, Pattern Recognition and Computer Vision. Automatic reading of characters in natural scenes is very useful to a wide range of applications such as autonomous vehicle navigation, textual image retrieval, machine translation, etc.

There are two widely used feature representations for scene character recognition including hand-crafted features and neural network based features. The majority of hand-crafted feature based methods employ HOG (Histogram of Oriented Gradient) [4] like features for character recognition, such as [25,30]. However, these traditional methods based on handcrafted features are not able to satisfactorily deal with noisy data in natural scenes. Recently, deep learning based models have been presented for solving character recognition problem. Convolutional Neural Networks (CNNs), which possess a powerful feature expression ability, are utilized to recognize characters in many works, such as [26,1,14,27].

Through our experiments, we find that existing CNN-based models typically fail in handling the following two situations:

- **Noisy background and texture.** CNNs employ convolutional filters to extract useful patterns and combine them for recognition. However, noisy background and texture tend to distract CNNs from locating useful patterns accurately.
- **Novel Font Design (or writing style).** The human creativity continually endows characters with novel font designs (or writing styles). As we know, even a small deformation of objects could lead off-the-shelf CNNs to failure in object recognition. Sim-

Y. Wang · Z. Lian · Y. Tang · J. Xiao
Institute of Computer Science and Technology, Peking University
E-mail: {wangyizhi, lianzhouhui, tangyingmin, xjg}@pku.edu.cn
Tel.: +86-10-82529245



Fig. 1 Scene character samples that existing CNN-based models typically fail to recognize. We classify the failure situations into two categories: one is noisy background and texture, the other is novel font design (or writing style).

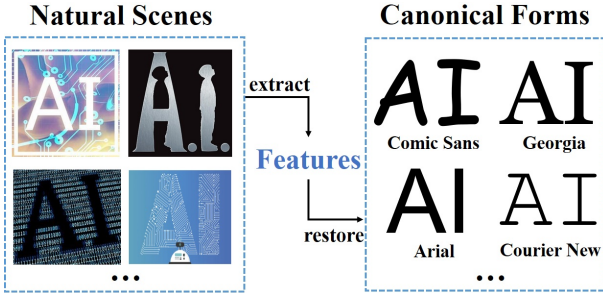


Fig. 2 The main idea of our work. Extracted features of scene characters are endowed with the ability to portray their canonical forms of glyphs in multiple font styles. By adding this constraint, redundant information in features is reduced and thus recognition performance can be improved.

ilarly, existing CNN-based models perform poorly in recognizing those new images whose font styles differ a lot with training images. Specific examples of these two situations are shown in Figure 1.

Existing methods ignore the following two important facts: First, glyphs in canonical forms can be adopted as helpful guidance for SCR in addition to character labels. Most existing models employ character labels as the only guidance information. In this paper, we define canonical forms of glyphs as the character images in some commonly-used and easy-to-read fonts, such as Arial, Times New Roman, Courier (English), Song and Kai (Chinese), etc. Furthermore, it is easy to obtain these character images as they can be directly generated from corresponding font files. Referring to canonical forms of glyphs is actually seeking for a mapping relation from scene characters to their standard glyphs. This mapping relation can instruct models to further exclude useless scene information and concentrate on

the shapes of characters. Moreover, we believe that this process is accordant with the habit of human cognition and behavior.

Second, font style information (letter endings, weights, slopes, etc.) is useless for recognizing character content and not supposed to be encoded into character features. Existing models tend to fail in recognizing character images with new font styles, which are commonly-seen in SCR scenarios while few previous works have paid attention to that. Accordingly, we define two important properties of features extracted from character images, which markedly affect the recognition performance:

- **Scene-independent.** Features for character recognition are supposed to focus on the characters’ shapes instead of scene information like background, illumination, special effects, etc. In our method, the extracted deep features are forced to be capable of reconstructing standard printed glyphs, which is helpful to reduce useless scene information. In this manner, our model acquires a better global perception of characters shapes so as to ignore noisy backgrounds and textures.
- **Font-independent.** Font information of characters ought not to be contained in extracted features for recognition. It is necessary to decrease the font style information in extracted features to increase the robustness of models. Our model is designed to generate glyphs of different font styles by decoupling content features and font style features. Therefore, the learnt features are supposed to contain as less font information as possible during training stage. When meeting characters with new font designs, our model could still find useful local patterns.

Motivated by the above-mentioned analyses, we propose a novel framework named Character Generation and Recognition Network (CGRN), in which canonical forms of glyphs are used as prior knowledge for guiding models to learn scene-independent and font-independent features for SCR.

2 Related Work

2.1 Image Synthesis

Image synthesizing methods essentially fall into two categories: parametric and nonparametric. The nonparametric models generate target images by copying patches from training images. In recent years, parametric models based on Generative Adversarial Networks (GANs) [7] have been popular and achieved impressive results, such as LAPGAN [6] and DCGAN [20]. Image-to-image translation is a specific problem in the area of

image synthesis. Most recent approaches utilize CNNs to learn a parametric translation function by training a dataset of input-output examples. Inspired by the success of GANs in generative tasks, the “pix2pix” framework [11] uses a conditional generative adversarial network to learn a mapping from input to output images. Our key observation is that the learned mapping from a complex distribution (natural scene) to a simple distribution (canonical form) can help to handle the recognition task. Motivated by this observation, generation and recognition functions are integrated together and work cooperatively in our model.

2.2 Scene Character Recognition

The problem of character recognition has attracted intensive attentions from AI/CV/PR communities for several decades. Traditional methods rely heavily on hand-crafted features (e.g., HOG [4]) and typically fail to obtain satisfactory performance on scene character recognition (SCR). Recently, deep learning based approaches became as the predominant way to handle the SCR task. For instance, [26], [1], [14] employed end-to-end CNNs to extract characters’ features for recognition. [27] proposed a model named SEDPD to improve the classification performance of CNN features by learning discriminative part detectors. In handwritten character recognition (HCR), [29] designed an adversarial feature learning (AFL) model to learn writer-independent features under the guidance of printed data. However, above mentioned methods are all classification-based feature learning frameworks while our model is based on generative models. Character labels are employed as their only guidance information in existing methods except AFL. Compared to AFL, our generative model aims to transfer the input scene character image to its canonical forms of glyphs, whose semantic and font information are both utilized by our method to excavate scene&font-independent features.

3 Method Description

In this section, we describe the methodology of learning canonical forms of glyphs and present the details of our proposed network (i.e., CGRN).

3.1 Character Generation and Recognition Network

CGRN builds a bridge between scene characters and standard glyphs to acquire a better understanding of their meanings. Specifically, given an input character

image x , CGRN tries to generate its corresponding glyphs of canonical forms in multiple font styles and predict its character label simultaneously. Let us denote its character label as y and its corresponding target images as $T = \{t_1, t_2, \dots, t_m\}$, where $y \in Y$, $Y = \{1, 2, \dots, L\}$ is a finite label set, L denotes the number of character classes, and m is the number of font categories.

3.2 Network Architecture

As shown in Figure 3, the proposed CGRN is composed of four subnetworks: Feature Extraction Network (FEN), Character Classification Network (CCN), Glyph Generating Network (GGN) and Glyph Discrimination Network (GDN). Details of the network architecture are presented in the following subsections.

3.2.1 Feature Extraction Network

The Feature Extraction Network is composed of a bunch of convolutional and pooling layers. Given the input character image x , FEN learns a deep feature $E(x, \theta_e)$, where θ_e denotes the parameters of FEN. To describe characters’ semantic information more precisely, multi-layer features are combined together to represent $E(x, \theta_e)$:

$$E(x, \theta_e) = \{E_{l_1}(x, \theta_e), E_{l_2}(x, \theta_e), \dots, E_{l_k}(x, \theta_e)\}, \quad (1)$$

where $E_{l_i}(x, \theta_e)$ denotes the output features of l_i -th layer ($l_1 < l_2 < \dots < l_k$).

3.2.2 Glyph Generating Network

The Glyph Generating Network takes the deep features of source images and font embeddings as input, and generates corresponding glyphs in canonical forms. Specifically, the extracted deep features are utilized to generate character images in multiple font styles through a bunch of deconvolutional layers, as shown in the “Multi-Font Output” part in Figure 3.

Given the extracted features $E(x, \theta_e)$ and a font embedding z_i , x ’s corresponding canonical form \hat{t}_i in font style f_i can be generated by GGN

$$\hat{t}_i = G(E(x, \theta_e), z_i, \theta_g), 1 \leq i \leq m, \quad (2)$$

where z_1, z_2, \dots, z_m are font category embedding vectors and θ_g denotes the parameters of GGN. The generation of $\hat{t}_1, \hat{t}_2, \dots, \hat{t}_m$ is parallel and the process is very similar to group convolution.

Inspired by the U-NET proposed by [21], we add skip connections in the corresponding layers of FEN and GGN to reduce the information loss during the downsampling process (shown in Figure 3).

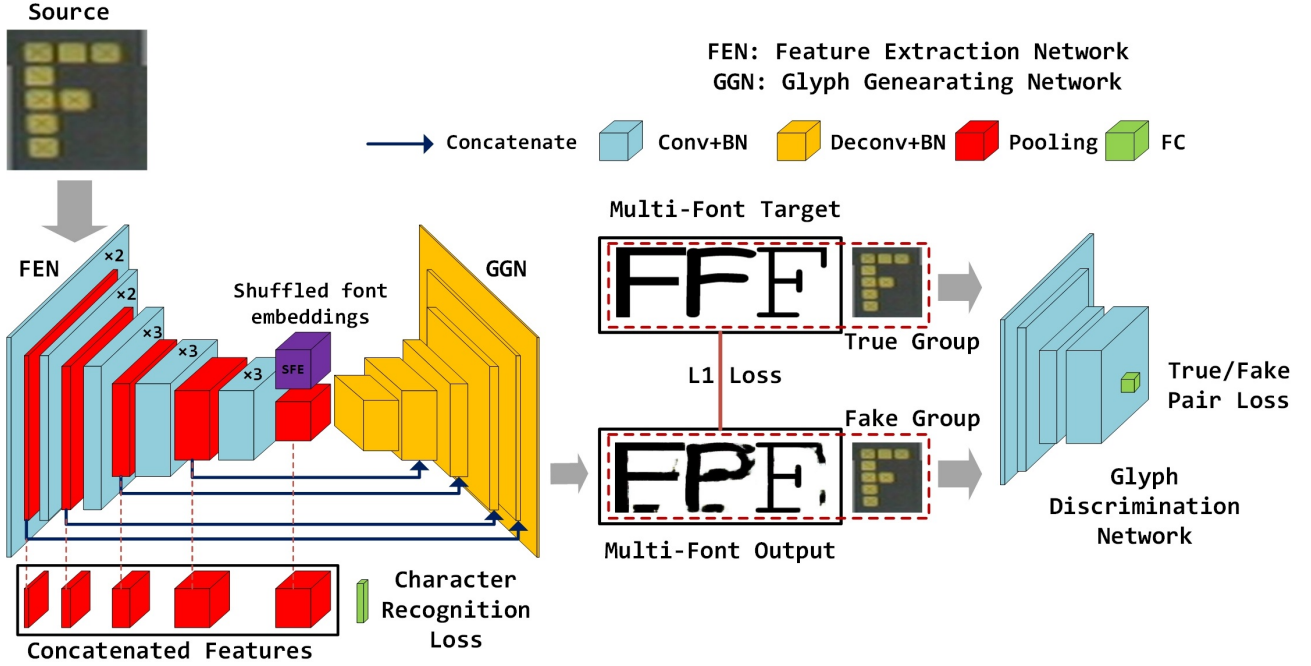


Fig. 3 The architecture of our proposed network.

3.2.3 Character Classification Network

The aim of our Character Classification Network (CCN) is to utilize the deep features extracted from input images to classify them into corresponding character categories. First, multi-scale features in $E(x, \theta_e)$ are down-sampled and concatenated as

$$E'(x, \theta_e) = \text{concat}(E'_{l_1}(x, \theta_e), \dots, E'_{l_k}(x, \theta_e)), \quad (3)$$

where $E'_{l_i}(x, \theta_e)$ is down-sampled from $E_{l_i}(x, \theta_e)$, $1 \leq i \leq k$. $E'(x, \theta_e)$ is then sent into CCN to compute its classification probability

$$P(y_c = y|x; \theta_e, \theta_c) = \frac{e^{-C_y(E'(x, \theta_e), \theta_c)}}{\sum_{j=1}^L e^{-C_j(E'(x, \theta_e), \theta_c)}}, \quad (4)$$

where y_c denotes the predicted category label, θ_c denotes the parameters of CCN, and $C_j(\bullet)$ means the value of j -th element in the CCN's output vector.

3.2.4 Glyph Discrimination Network

The introduction of Glyph Discrimination Network (GDN) mainly borrows the idea of Generative Adversarial Networks (GANs) [7]. GAN has been proved to be able to markedly improve the quality of generated images. In CGRN, GDN is employed to discriminate between generated (fake) glyphs and real glyphs. Meanwhile, GGN is devoted to generate glyphs which are good enough to fool GDN. Through this adversarial process, the quality of generated glyphs becomes better and better.

GDN takes an image pair as input and predict its label y_d . The real pair (x, t) corresponds to $y_d = 1$, while the fake pair (x, \hat{t}) corresponds to $y_d = 0$, where $t \in T = \{t_1, t_2, \dots, t_m\}$, $\hat{t} \in \hat{T} = \{\hat{t}_1, \hat{t}_2, \dots, \hat{t}_m\}$. The probabilities of $y_d = 1$ and $y_d = 0$ predicted by GDN are defined as

$$P(y_d = 1|x, t_i; \theta_d) = \frac{1}{1 + e^{-D(x, t_i, \theta_d)}}, \quad (5)$$

$$P(y_d = 0|x, t_i; \theta_d) = 1 - P(y_d = 1|x, t_i; \theta_d), \quad (6)$$

$$P(y_d = 1|x, \hat{t}_i; \theta_d) = \frac{1}{1 + e^{-D(x, \hat{t}_i, \theta_d)}}, \quad (7)$$

$$P(y_d = 0|x, \hat{t}_i; \theta_d) = 1 - P(y_d = 1|x, \hat{t}_i; \theta_d), \quad (8)$$

where θ_d denotes the parameters of GDN, $1 \leq i \leq m$ and $D(\bullet)$ means the value of GDN's output.

3.2.5 Detailed Configurations

The detailed configurations of CGRN are shown in Table 1. The first, second and third columns show network layers' names, types and parameters, respectively. Thereinto, "Conv" means convolutional layer, "Pool" means pooling layer, "Deconv" means deconvolutional

layer and “FC” means fully-connected layer. For convolutional and deconvolutional layers, “ $k \times k \times c, s, \text{BN}, \text{ReLU}$ ” in each row denotes that the kernel size is k , the stride is s , the number of output features’ channels is c , activation function ReLU and batch normalization [10] are employed. For pooling layers, “ $k \times k, s$ ” in each row denotes that the pooling kernel size is k , the stride is s . For fully-connected layers, “ $i \times o$ ” in each row describes the input dimension and output dimension of features. The configuration of FEN is basically consistent with convolutional layers of VGG16 [24], except the last pooling layer and batch normalization for each convolutional layer. Note that the architectures of our FEN and GGN are asymmetrical, which is different against the original UNET, as we expect FEN goes deeper to acquire richer features of scene characters.

Table 1 Detailed Configurations of our CGRN.

Input: 64×64 RGB image		
Layer	Type	Parameters
Feature Extraction Network		
$E_conv1 (\times 2)$	Conv	$3 \times 3 \times 64, 1, \text{BN}, \text{ReLU}$
E_pool1	Pooling	$2 \times 2, 2$
$E_conv2 (\times 2)$	Conv	$3 \times 3 \times 128, 1, \text{BN}, \text{ReLU}$
E_pool2	Pooling	$2 \times 2, 2$
$E_conv3 (\times 3)$	Conv	$3 \times 3 \times 256, 1, \text{BN}, \text{ReLU}$
E_pool3	Pooling	$2 \times 2, 2$
$E_conv4 (\times 3)$	Conv	$3 \times 3 \times 512, 1, \text{BN}, \text{ReLU}$
E_pool4	Pooling	$2 \times 2, 2$
$E_conv5 (\times 3)$	Conv	$3 \times 3 \times 512, 1, \text{BN}, \text{ReLU}$
E_pool5	Pooling	$4 \times 4, 4$
Character Classification Network		
C_pool1	Pooling	$32 \times 32, 32$
C_pool2	Pooling	$16 \times 16, 16$
C_pool3	Pooling	$8 \times 8, 8$
C_pool4	Pooling	$4 \times 4, 4$
C_pool5	Pooling	$1 \times 1, 1$
C_fc	FC	$1472 \times L$
Glyph Generating Network		
$G_deconv1,2$	Deconv	$5 \times 5 \times 512, 2, \text{BN}, \text{ReLU}$
$G_deconv3$	Deconv	$5 \times 5 \times 256, 2, \text{BN}, \text{ReLU}$
$G_deconv4$	Deconv	$5 \times 5 \times 128, 2, \text{BN}, \text{ReLU}$
$G_deconv5$	Deconv	$5 \times 5 \times 64, 2, \text{BN}, \text{ReLU}$
$G_deconv6$	Deconv	$5 \times 5 \times 3, 2, \text{BN}, \text{ReLU}$
Glyph Discrimination Network		
D_conv1	Conv	$5 \times 5 \times 64, 2, \text{BN}, \text{ReLU}$
D_conv2	Conv	$5 \times 5 \times 128, 2, \text{BN}, \text{ReLU}$
D_conv3	Conv	$5 \times 5 \times 256, 2, \text{BN}, \text{ReLU}$
D_conv4	Conv	$5 \times 5 \times 512, 1, \text{BN}, \text{ReLU}$
D_fc	FC	$32, 768 \times 1$

3.3 Learning Scene&Font-Independent Features

Although CNN based models have already achieved impressive performance in removing useless scene information for character recognition, we believe their performance can be further improved by adding the guidance of glyphs in canonical forms. As shown in Figure 3, the target images are composed of clear backgrounds and corresponding glyphs in canonical forms. The aim of the proposed CGRN is to learn scene&font-independent

features by reconstructing these character images accurately.

As we know, glyphs in any kinds of font styles might be contained in a given natural scene image. If we mix semantic features with font information, our model might fail to recognize those characters whose font styles have never been seen in training images. Therefore, the extracted character features $E(x, \theta_e)$ are supposed to be font-independent in addition to scene-independent. By introducing the font embedding mechanism into CGRN, we expect to separate font features from semantic features, with z representing the former and $E(x, \theta_e)$ for the latter. Furthermore, by setting multi-font glyphs as target, the extracted features are forced to contain as less font information as possible, so as to reconstruct glyphs in various font styles. However, if we set single-font glyphs as target, font information is hardly to be removed especially when many training characters’ fonts share similar style with the target font. In our work, we select English and Chinese characters in representative fonts as the target glyphs in canonical forms for CGRN to generate, details are presented in the following sections.

3.4 Loss Function

In CGRN, we define three loss functions. The first one (i.e., the pixel loss L_{pixel}), which measures the dissimilarity between generated character images \hat{t} and their corresponding glyphs of canonical forms t all over m fonts, is defined as

$$\begin{aligned}
 L_{pixel} &= \mathbb{E}_{t, \hat{t}} \left[\frac{1}{m} \sum_{i=1}^m \|\hat{t}_i - t_i\| \right] \\
 &= \mathbb{E}_{x, t, z} \left[\frac{1}{m} \sum_{i=1}^m \|G(E(x, \theta_e), z_i, \theta_g) - t_i\| \right].
 \end{aligned} \tag{9}$$

The character recognition loss L_{CR} , which indicates the recognition error of CCN, is defined as

$$L_{CR} = -\mathbb{E}_{x, y} [\log(P(y_c = y | x; \theta_e, \theta_c))]. \tag{10}$$

The discriminator loss L_D , which indicates the identification error of GDN all over m image pairs, is defined as

$$\begin{aligned}
 L_D &= -\mathbb{E}_{x, t} \left[\frac{1}{m} \sum_{i=1}^m \log(P(y_d = 1 | x, t_i; \theta_d)) \right] \\
 &\quad - \mathbb{E}_{x, \hat{t}} \left[\frac{1}{m} \sum_{i=1}^m \log(P(y_d = 0 | x, \hat{t}_i; \theta_d)) \right].
 \end{aligned} \tag{11}$$

3.5 Training Process

We optimize the whole network by introducing GDN to play the minmax game with FEN, GGN, and CCN:

$$\min_{\theta_e, \theta_c, \theta_g} \max_{\theta_d} \lambda L_{CR}(\theta_e, \theta_c) + \lambda L_{pixel}(\theta_e, \theta_g) - L_D(\theta_e, \theta_g, \theta_d), \quad (12)$$

where λ is a fixed weight coefficient (set as 100 in our experiments). Our optimization strategy for this objective function is training FEN, CCN, GGN and GDN alternatively, which is similar to [7]. Details are as follows.

In each mini-batch, we first optimize GDN:

$$\theta_d \leftarrow \theta_d - \mu \frac{\partial L_D}{\partial \theta_d}. \quad (13)$$

Then, we optimize FEN, GGN and CCN jointly:

$$\theta'_e \leftarrow \theta_e - \mu \left(\frac{\partial \lambda L_{CR}}{\partial \theta_e} + \frac{\partial \lambda L_{pixel}}{\partial \theta_e} - \frac{\partial L_D}{\partial \theta_e} \right), \quad (14)$$

$$\theta'_g \leftarrow \theta_g - \mu \left(\frac{\partial \lambda L_{pixel}}{\partial \theta_g} - \frac{\partial L_D}{\partial \theta_g} \right), \quad (15)$$

$$\theta'_c \leftarrow \theta_c - \mu \frac{\partial \lambda L_{CR}}{\partial \theta_c}, \quad (16)$$

$$\theta_e \leftarrow \theta'_e, \theta_g \leftarrow \theta'_g, \theta_c \leftarrow \theta'_c, \quad (17)$$

where μ is the learning rate used for training model. Note that Equation 13, 14, 15 and 16 are presented in the basic form of gradient descent for brevity. Practically, we employ a gradient-based optimizer with adaptive moment estimation (i.e., Adam optimizer [16]).

4 Experiments

To verify the effectiveness and generality of our model, we conduct experiments on datasets of three widely-used languages: English, Chinese and Bengali, which represent alphabet, logography and abugida, respectively. We also design comparative experiments to demonstrate the effect of proposed techniques in our model.

4.1 Datasets

English character datasets include **ICDAR 2003** [18] and **IIIT5K** [19] dataset. They all contain English letters in 52 classes and Arabic numbers in 10 classes (i.e., a-z, A-Z, 0-9).

- The task on the ICDAR 2003 character dataset is very challenging because of serious non-text background outliers with cropped character samples, and many character images have very low resolution. It contains 6,113 character images for training and 5,379 for testing
- The IIIT5K dataset consists of 9,678 character samples for training and 15,269 for testing. The dataset contains both scene text images and born-digital images.

Chinese character dataset : **Pan+ChiPhoto** dataset [25]. It is built by the combination of two datasets: **ChiPhoto** and **Pan_Chinese_Character** dataset. The images in this dataset are mainly captured at outdoors in Beijing and Shanghai, China, which involve various scenes like signs, boards, advertisements, banners, objects with texts printed on their surfaces. In [25], ChiPhoto was split into two parts (60% and 40%) and added to the training and testing datasets of Pan_Chinese_Character. Because the authors did not give the exact split of training and testing images, we follow their steps to split these images again. Finally, we have 6098 training images and 4220 testing images, totally 1203 character classes.

Bengali character dataset: **ISI_Bengali_Character** dataset [25]. Bengali script is the 6th most popularly used script in the world and it holds official status in the two neighboring countries Bangladesh and India. This dataset contains 158 classes of Bengali numerals, characters or their parts. 19,530 Bengali character samples are divided into two sets in which 13,410 images are used for training and 6,120 for testing.

Selected fonts for target canonical forms of glyphs: In our experiments, we select 4 fonts for English glyphs: **Arial**, **Comic Sans**, **Courier New** and **Georgia**, 4 fonts for Chinese glyphs: **Song**, **Kai**, **Hei** and **Fangsong**, and one font for Bengali glyphs: **Nirmala UI**. Character samples rendered by these fonts are shown in Fig 4. For English and Chinese, we select these fonts based on their popularity and style diversity. We only select one font for Bengali glyphs because we are not familiar with the language.

4.2 Data Preprocessing

All character images (including source images and target images) are resized into the resolution of 64×64 .

Font name	Character samples
Arial	Character Recognition
Comic Sans	Character Recognition
Courier New	Character Recognition
Georgia	Character Recognition
Song	字符识别
Kai	字符识别
Hei	字符识别
Fangsong	字符识别
Nirmala UI	অক্ষর স্বীকৃতি

Fig. 4 Samples of characters rendered by our selected fonts.

Besides shuffling training data images, font embeddings are also shuffled for each training image x to avoid overfitting: $(z_1, z_2, \dots, z_m) \rightarrow (z_{p_1}, z_{p_2}, \dots, z_{p_m})$, where p_1, p_2, \dots, p_m is one permutation of $1, 2, \dots, m$. Accordingly, the target images (t_1, t_2, \dots, t_m) are shuffled to $(t_{p_1}, t_{p_2}, \dots, t_{p_m})$.

4.3 Implementation Details

The parameters of FEN are initialized by the pre-trained VGG16 [24] on ImageNet dataset [5], and the others are initialized by using random weights with normal distribution of 0 mean and 0.02 standard deviation. We adopt Adam optimizer [16] to train our model where the initial learning rate μ is set to 0.0001 and the first order of momentum is 0.5. The batch size of training images is set to 128. Our method is implemented under the TensorFlow framework [9].

Table 2 Character recognition results of different methods evaluated on the English Datasets

Method	IC03	IIIT5K
HOG + SVM [4]	77.0	70.0
CNN [14]	86.8	78.5
RPCA-SR [30]	79.0	76.0
CO-HOG [25]	80.5	77.8
ConvCoHOG [25]	81.7	78.8
SEDPD [27]	84.0	80.3
VGG16 (pretrained)	84.5	82.1
CGRN (-GGN, -GDN)	85.5	84.9
CGRN (SF, -GDN)	86.3	85.2
CGRN (MF, -GDN)	86.9	85.5
CGRN (SF)	86.8	85.5
CGRN (MF)	87.1	85.6

Table 3 Character recognition results of different methods evaluated on the Chinese Dataset

Method	Pan+ChiPhoto
Tesseract OCR [8]	20.5
HOG	59.2
CNN [14]	61.5
CNN +SVM [14]	62.3
ConvCoHOG_NewOffset [25]	71.2
VGG16 (pretrained)	85.8
CGRN (-GGN, -GDN)	87.5
CGRN (SF, -GDN)	88.6
CGRN (MF, -GDN)	88.9
CGRN (SF)	89.1
CGRN (MF)	89.4

Table 4 Character recognition results of different methods evaluated on the Bengali Dataset

Method	ISI.Bengali
Tesseract OCR [8]	-
HOG	87.4
CNN [14]	89.7
CNN+SVM [14]	90.1
ConvCoHOG_NewOffset [25]	92.2
VGG16 (pretrained)	96.3
CGRN (-GGN, -GDN)	96.9
CGRN (-GDN)	97.1
CGRN	97.4

4.4 Comparison with Other Methods

Table 2, 3, and 4 show different methods' performance on aforementioned datasets. As the parameters of FEN are initialized by the pre-trained VGG16, we add a fine-tuned VGG16 model into comparison for the sake of fairness. In these tables, "SF" denotes the best performing version of CGRN when employing single-font glyphs of canonical forms as target. "-GDN" denotes the CGRN model in which GDN is removed. "-GGN" denotes the CGRN model in which GGN is removed. "MF" means employing multi-font glyphs as target in CGRN (we experimentally set $t = 4$ in English and Chinese datasets). As shown in these tables, our method clearly outperforms other existing approaches.

For the experiments conducted on IC03 (ICDAR 2003), [14] and [25] both used additional character images (107k and 18k) to train their models. On the contrary, our model uses the least training character images (6k) and achieves the best performance (see Table 2). Although utilizing the parameters of pre-trained VGG16, our model still possesses the advantages of high efficiency and better extensibility. As is known to all, recognizing Chinese characters is a very challenging task on account of their large number of classes and quite complicated geometric structures. Existing methods perform poorly on the Pan+ChiPhoto dataset. VGG16 network greatly improves the recognition accu-

racy, owing to the pre-training on the ImageNet dataset. By learning canonical forms of glyphs, our model is able to extract more discriminative features for Chinese characters in natural scene images and thus significantly improves the recognition accuracy (see Table 3). ISI_Bengali_Character, which contains many synthesized images, is a less challenging dataset compared to the others. Our model mainly contributes to improving the recognition accuracy of scene characters in this dataset, so the improvement of the overall recognition accuracy is not that significant (see Table 4).

4.5 The Improvement of Image Feature Learning

In this section we give an illustration of how the feature learning is improved by learning canonical forms of glyphs. We select VGG16 network for comparison, which is commonly used as feature extractor in many existing models [13, 22, 17]. Utilizing the technique proposed by [28], we project the features activations back to the input pixel space, which is presented in Figure 5. The light regions in the reconstructed images are responsible for the activations in extracted feature maps. The selected cases in Figure 5 are representative of the two situations mentioned in Section 1. The left four cases possess novel font designs and the right four cases are placed in noisy backgrounds or have noisy textures. Compared to VGG16, our model shows more accurate perception on the discriminative patterns of scene characters and correctly recognize them. Our model is trained to capture the most discriminative local patterns so as to reconstruct multi-font canonical glyphs, which is the key to success. Taking the letter “E” for example, our model pays more attention to the middle horizontal line of “E”, which avoids recognizing it as “G”. Another example is the letter “f”, our model’s concentration on the head part of “f” leads to the successful recognition. Learning to generate canonical forms of glyphs instructs our model to exclude undesired nuisance factors and concentrate on those useful patterns.

4.6 Contribution of Glyphs Discrimination Network

Glyphs Discrimination Network is proved to be effective in improving the quality of generated glyphs through our experiments (see Figure 6). Without the supervision of GDN, FEN and GGN tend to generate incorrect or unrecognizable glyphs when meeting blurry, distorted and ambiguous character images. After introducing GDN, the generated images become more accordant with target images and easier to be recognized, although there still exist some flaws on them.

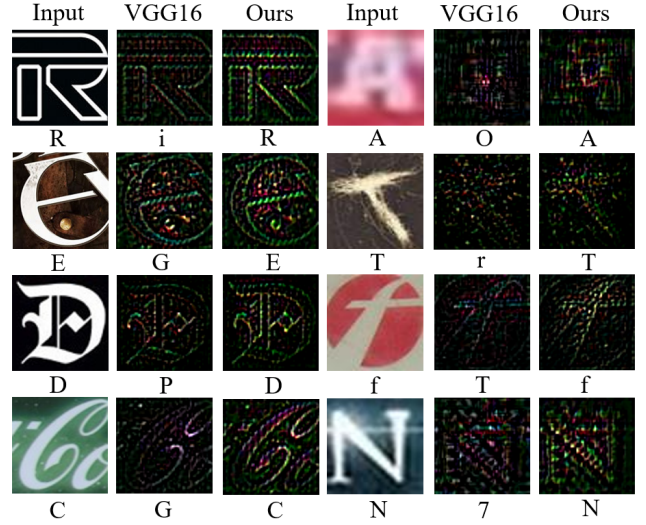


Fig. 5 Visualization of CNN feature maps of the 5th pooling layer from VGG16 and our model. The light pixels reveal the structures of input image that excite CNN feature maps. The characters below each set of images are ground-truth label, VGG16’s predicted label and our model’s predicted label, respectively. Our model captures more useful local patterns of scene characters compared to VGG16.

The improvement demonstrates that features learned by model with GDN are able to more precisely capture semantics and eliminate other interference information. Consequently, the recognition accuracy is also improved (shown in Table 2, 3, and 4). Character images in Figure 6 were wrongly classified without GDN but correctly classified after introducing GDN. Through improving the quality of generated images, the learned deep features are equipped with enhanced expressiveness and become more scene-independent.

4.7 Contribution of Multiple-Font Guidance

Our model becomes less sensitive to font variance under the multiple-font guidance (MFG) compared to the single-font guidance (SFG). When we set multi-font glyphs as generation targets, there must be at least one glyph whose font style is different from the input scene character. To precisely reconstruct them all, the feature extractor must learn to capture the most discriminative local patterns for every training example. By contrast, the training process of single-font model is not so sufficient as multi-font model, which leads single-font model more sensitive to the font variance. When we adopt a single font as the target, our model tends to generate incorrect glyphs if the font style of input character images differs a lot with the target font style, as is shown in Figure 7. It is caused by the poor robustness of extracted features mixed with a lot of unneces-



Fig. 6 Samples of testing images which were wrongly classified without GDN but correctly classified after introducing GDN. Glyphs in the “*output**” and “*output*” column are generated by CGRN (no GDN) and CGRN respectively.

sary font information. But the character images in Figure 7 can be correctly classified when training our model with multiple-font guidance. Multiple-font guidance instructs our model to distinguish semantics from font styles. Table 2, 3, 4 and Figure 7 show that MFG further enhances our model’s recognition and generation performance. Our model understands characters more deeply by reconstructing them into canonical forms of glyphs in multiple fonts according to font embeddings.

4.8 Experiments on Text String Recognition

The latest scene text recognition methods [22, 2, 3, 23] are mainly designed to recognize text strings without explicitly splitting characters by combining Recurrent Neural Networks (RNNs) with CNNs. To further verify our methods effectiveness, we conduct experiments on text string recognition in addition to single character recognition. Specifically, we add the proposed glyph generating and discrimination network to ASTER proposed by Shi *et al.* [23]. The detailed architecture of this new model, named as ASTER+CGN (Character Gen-

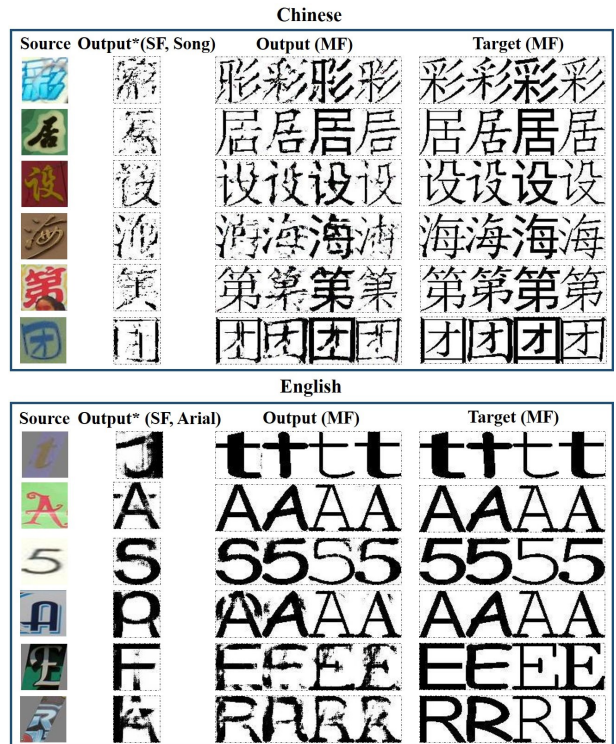


Fig. 7 Samples of testing images which were wrongly classified without MFG but correctly classified after introducing MFG. Glyphs in the “*output**” and “*output*” column are generated by CGRN (SF, GDN) and CGRN (MF, GDN) respectively.

eration Network), is presented in Figure 8. ASTER consists of a text rectification network and a text recognition network, corresponding to the blue and green modules in Figure 8, respectively. The newly added modules by us (marked as orange) consist of a LSTM encoder (for sequence modeling), a glyph generating network (i.e., glyph generator in Figure 8) and a glyph discrimination network. We employ the extracted CNN features to generate canonical forms of a horizontal text, following the main idea of CGRN. We add another dataset ICDAR-2013 [15] into the experiment. As shown in Table 5, our method significantly improves the recognition accuracy of ASTER and achieves state-of-the-art performance. Considering that the benefits of CGN have been fully discussed in previous sections, we will not show more experimental results here.

5 Towards Common Object Recognition

In this paper, our work focuses on scene character recognition in natural scenes. It is worth mentioning that the main idea of our work is not only limited to text recognition but also applicable to common object recognition tasks, such as face recognition with different poses,

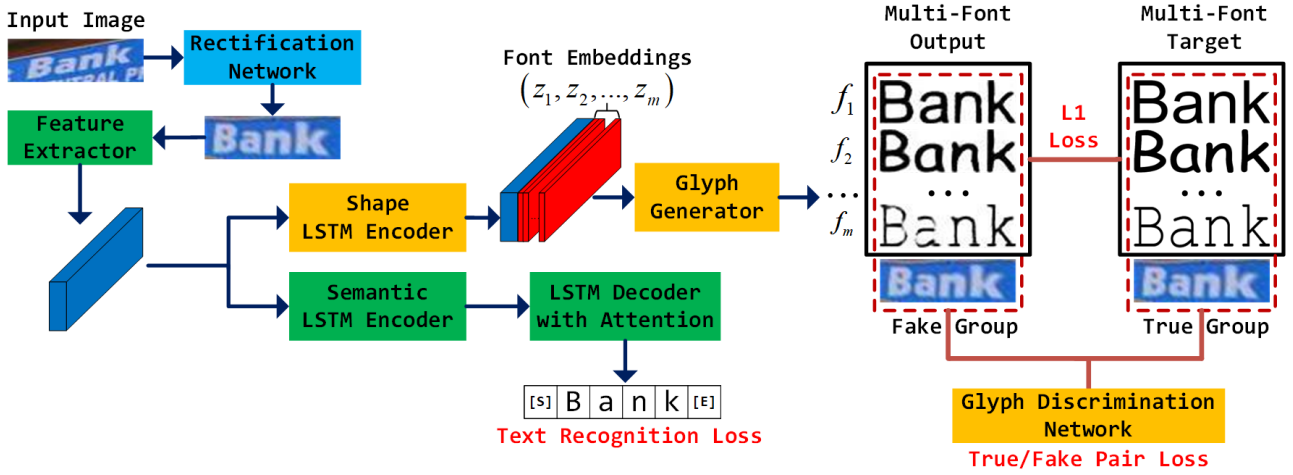


Fig. 8 The architecture of our proposed network (ASTER+CGN) for text string recognition.

Table 5 Text string recognition results of different methods evaluated on benchmarks in lexicon-free mode.

Method	IC03	IIIT5K	IC13
Jaderberg <i>et al.</i> [12]	89.6	-	81.8
Jaderberg <i>et al.</i> [13]	93.1	-	90.8
Lee <i>et al.</i> [17]	88.7	78.4	90.0
Shi <i>et al.</i> [22]	91.9	81.2	89.6
Cheng <i>et al.</i> [2]	94.2	87.4	93.3
Cheng <i>et al.</i> [3]	91.5	87.0	-
Shi <i>et al.</i> [23]	94.5	93.4	91.8
ASTER+CGN	95.3	94.0	94.4

speech recognition with accents and so on. Canonical forms of objects, which contribute to eliminate the interference factors in scene images, are more meaningful than any other object properties summarized by human. Therefore, we believe that canonical forms of objects could be used as more complete labels than traditional numeric labels in many applications of object recognition.

6 Conclusion

This paper presented a deep learning based framework to boost the performance of SCR by learning canonical forms of glyphs. The main contribution of the paper is to utilize canonical forms of glyphs as guidance to excavate scene-independent and font-independent features. Our model also benefits from the adversarial learning process introduced in GAN. The effectiveness and generality of our techniques were verified through experiments conducted on multilingual datasets. Last but not the least, it is easy to extend our approach with a few changes to improve the performance of methods in many other object recognition tasks.

References

1. Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: Photoocr: Reading text in uncontrolled conditions. In: IEEE International Conference on Computer Vision, pp. 785–792 (2013)
2. Cheng, Z., Bai, F., Xu, Y., Zheng, G., Pu, S., Zhou, S.: Focusing attention: Towards accurate text recognition in natural images. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 5086–5094 (2017)
3. Cheng, Z., Xu, Y., Bai, F., Niu, Y., Pu, S., Zhou, S.: Aon: Towards arbitrarily-oriented text recognition. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5571–5579 (2018)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision & Pattern Recognition, pp. 886–893 (2005)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 248–255 (2009)
6. Denton, E.L., Chintala, S., Szlam, A., Fergus, R.: Deep generative image models using a laplacian pyramid of adversarial networks. neural information processing systems pp. 1486–1494 (2015)
7. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Advances in Neural Information Processing Systems **3**, 2672–2680 (2014)
8. Google: Tesseract optical character recognition. <https://code.google.com/p/tesseract-ocr/> (2006)
9. Google: Tensorflow. <https://www.tensorflow.org/> (2016)
10. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015)
11. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5967–5976 (2017)
12. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Deep structured output learning for unconstrained text recognition. international conference on learning representations (2015)

13. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision* **116**(1), 1–20 (2016)
14. Jaderberg, M., Vedaldi, A., Zisserman, A.: Deep features for text spotting. In: *European Conference on Computer Vision*, pp. 512–528 (2014)
15. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazn, J.A., de las Heras, L.P.: Icdar 2013 robust reading competition. In: *2013 12th International Conference on Document Analysis and Recognition*, pp. 1484–1493 (2013)
16. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. *Computer Science* (2014)
17. Lee, C.Y., Osindero, S.: Recursive recurrent nets with attention modeling for ocr in the wild. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2231–2239 (2016)
18. Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: Icdar 2003 robust reading competitions. *Proc of the Icdar* **7**(2-3), 105–122 (2003)
19. Mishra, A., Alahari, K., Jawahar, C.V.: Scene text recognition using higher order language priors (2013)
20. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. *Computer Science* (2015)
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241 (2015)
22. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **39**(11), 2298–2304 (2017)
23. Shi, B., Yang, M., Wang, X., Lyu, P., Yao, C., Bai, X.: Aster: An attentional scene text recognizer with flexible rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* pp. 1–1 (2018)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *Computer Science* (2014)
25. Tian, S., Bhattacharya, U., Lu, S., Su, B., Wang, Q., Wei, X., Lu, Y., Tan, C.L.: Multilingual scene character recognition with co-occurrence of histogram of oriented gradients. *Pattern Recognition* **51**(C), 125–134 (2016)
26. Wang, T., Wu, D.J., Coates, A., Ng, A.Y.: End-to-end text recognition with convolutional neural networks. In: *International Conference on Pattern Recognition*, pp. 3304–3308 (2013)
27. Wang, Y., Shi, C., Xiao, B., Wang, C.: Learning spatially embedded discriminative part detectors for scene character recognition. In: *Iapr International Conference on Document Analysis and Recognition*, pp. 363–368 (2017)
28. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. *European conference on computer vision* pp. 818–833 (2014)
29. Zhang, Y., Liang, S., Nie, S., Liu, W., Peng, S.: Robust offline handwritten character recognition through exploring writer-independent features under the guidance of printed data. *Pattern Recognition Letters* **106** (2018)
30. Zhang, Z., Xu, Y., Liu, C.L.: Natural scene character recognition using robust pca and sparse representation. In: *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*, pp. 340–345. *IEEE* (2016)