RESEARCH ARTICLE

Text Localization and Recognition of Chinese Characters in Natural Scenes Based on Improved Faster Region-Based Convolutional Neural Network

Yuejie Li[1,*], **Shijun Li**[2] **and Zhendong Luo**[3,*]

[1] School of Mathematics and Computer Engineering, Ordos Institute of Technology, Ordos 017000, China
[2] Hunan Institute of Engineering, Xiangtan 411104, China
[3] School of Digital Intelligence, Nanchang Polytechnic University, Nanchang 330500, China

Abstract

To solve the problems faced by Chinese character recognition, such as complex shapes and diverse structures, this paper adopts the Visual Geometry Group 16 (VGG-16) model for feature extraction and introduces a two-layer bidirectional Long Short-Term Memory (LSTM) network. It improves the Faster Region-based Convolutional Neural Network (Faster R-CNN) by using a Region Proposal Network (RPN) to extract candidate boxes and adjust the positions of candidate regions. The improved model, namely Faster BLSTM-CNN, is tested through three types of experiments: validation of feature extraction effectiveness, comparative analysis of the algorithm before and after improvement, and comparison with traditional recognition algorithms. Finally, an experimental comparison combining text recognition and localization is conducted. On the ReCTS and MSRA-TD500 datasets, the proposed Faster BLSTM-CNN achieves excellent performance in Chinese character localization and recognition: in natural scenes (ReCTS dataset), its total image identification rate (TIIR) reaches 81.54%, recognition precision rate is 88.14%, and inference speed is 86 ms per image. Compared with the end-to-end benchmark algorithms DETR+CRNN-RMC and CTPN+CRNN-RMC, it improves the recognition precision rate by up to 7.54%, the TIIR by up to 10.04%, and the inference speed by up to 21%. These gains are mainly attributed to the two-layer bidirectional LSTM for contextual feature modeling and the element-wise addition fusion strategy that avoids dimension explosion.

Keywords: text localization and recognition, deep learning, faster region-based convolutional neural network, long short-term memory.

Submitted: 23 April 2026
Revised: 22 August 2026
Accepted: 22 September 2026
Published: 27 September 2026

Vol. 3, **No.** 3, 2026.
10.62762/TETAI.2026.620822

***Corresponding authors:**
✉ Yuejie Li
lyj@oit.edu.cn
✉ Zhendong Luo
zhdluo@ncepu.edu.cn

Citation
Li, Y., Li, S., & Luo, Z. (2026). Text Localization and Recognition of Chinese Characters in Natural Scenes Based on Improved Faster Region-Based Convolutional Neural Network. *ICCK Transactions on Emerging Topics in Artificial Intelligence*, 3(3), 202–217.

1 Introduction

Text retrieval from natural environments steadily ranks as a critical research topic in computer science. As contemporary mobile and multimedia terminals undergo swift evolution, an expanding need has emerged for identifying textual content in natural scene visuals. Across various practical scenarios, such text identification capability can underpin educational programs and instant translation solutions efficiently. It may also aid smart driving platforms in recognizing traffic markers to gather core roadway details for intelligent obstacle avoidance, and it can also assist embassies in document entry, cataloging, and indexing to realize digitization. Additionally, it enables network image and video retrieval systems to use text annotations in images for detection, thereby efficiently searching for target images. All these facts illustrate that text identification is absolutely necessary. This technology provides tangible convenience for current users' daily work and possesses notable scholarly research significance. However, Chinese text differs from English text, which is composed of combinations of 26 letters. Comprehensive databases of Chinese characters exhibit sophisticated typeface structures and varied form expressions, creating considerable hurdles for the localization and recognition of Chinese text content. Therefore, under the same background interference, the inherent complexity of Chinese characters leads to greater difficulty in recognizing and localizing Chinese text within real-world application scenarios.

In particular, the complexity of Chinese characters (e.g., diverse glyph structures, vertical/horizontal mixed layouts, and subtle differences between similar characters) poses unique challenges to existing object detection frameworks. For Faster R-CNN, which was originally designed for general object detection, its lack of contextual sequence modeling leads to two critical issues in Chinese text tasks: first, it fails to capture the dependencies between consecutive characters, resulting in misrecognition of similar glyphs; second, it cannot handle irregular text layouts (e.g., inclined or curved text lines), leading to incomplete text line detection.

As text recognition application scenarios expand steadily, relevant researchers keep promoting in-depth exploration within this specialized field. Kantipudi et al. [1] proposed a scene text recognition approach integrating bidirectional LSTM and deep CNN, where contour detection is first applied to the image before feeding it into the CNN for feature sequence extraction, achieving high recognition accuracy on the MSRA-TD500 dataset. However, their method focuses solely on recognition without addressing text localization in complex natural scenes. Ye and Doermann [2] provided a comprehensive survey of text detection and recognition methods in natural scene imagery, categorizing approaches by detection strategy and recognition pipeline. Although this survey offers a thorough overview, most reviewed methods focus on English text and do not specifically address the challenges of complex glyph structures inherent to Chinese characters. Raisi et al. [3] utilized a Transformer-based architecture for text image recognition, which is more efficient than previous methods. However, its advantages are limited, and local features are easily lost. Notably, most of the above studies focus on English text recognition and cannot be directly applied to Chinese characters, which are characterized by complex glyph structures, irregular layout in natural scenes, and severe interference from complex backgrounds. These unique challenges make Chinese scene text recognition more difficult than English text recognition, especially in the integration of text localization and sequence recognition tasks. To address this gap, Yao et al. [4] proposed a unified framework for multioriented text detection and recognition in natural scenes and introduced the MSRA-TD500 dataset, which supports Chinese and English text at arbitrary orientations. However, the method relies on hand-crafted features and lacks the deep contextual modeling capabilities required for recognizing complex Chinese characters. Tong et al. [5] proposed MA-CRNN, a multi-scale attention CRNN for Chinese text line recognition in natural scenes, which enhances feature extraction for Chinese characters of diverse types and complex structures. However, it does not address the joint localization and recognition of Chinese characters in complex natural scene images, nor does it explicitly model the contextual dependencies between consecutive characters. The large number of Chinese characters and their inherent similarity in glyphs still pose great challenges to recognition, requiring further optimization.

To solve the problems of complex graphic elements and indistinguishability from background scenes in Chinese character recognition, many researchers have introduced Faster R-CNN for image recognition. Ren et al. [6] merged the RPN and Fast R-CNN into a single network by sharing convolutional features, in which the RPN acts as an "attention mechanism

that tells the detector where to look." This method greatly improves detection speed, but, as a general object detector, it does not model the sequential context of text. Long et al. [7] systematically reviewed the progress of scene text detection and recognition in the deep learning era, highlighting that while region-based methods such as Faster R-CNN have achieved strong detection performance, they generally lack contextual sequence modeling capabilities essential for recognizing complex scripts. Liao et al. [8] proposed a real-time scene text detection method based on differentiable binarization (DB), which simplifies post-processing and significantly improves detection speed and accuracy on standard benchmarks including MSRA-TD500. However, as a detection-only method, it does not address the joint localization and recognition of Chinese characters. Sri et al. [9] proposed an improved Faster R-CNN for image recognition and localization, achieving better accuracy and mean average precision. These studies show that Faster R-CNN has high recognition accuracy and the ability to distinguish complex scenes, but it lacks the ability to model text contextual correlations–a key limitation for Chinese character recognition (e.g., incomplete text line detection).

Based on the above challenges of Chinese text recognition and the pros and cons of Faster R-CNN, this paper proposes an improved Faster R-CNN and applies it to Chinese character localization and recognition. The improvements include: using the VGG-16 model (a CNN-based backbone) for feature extraction, introducing a two-layer bidirectional LSTM network (existing studies mostly use single-layer LSTM, which cannot fully capture bidirectional contextual information), and using RPN to extract and adjust candidate regions. The innovation of this study lies in the integration of a two-layer bidirectional LSTM network, which directly addresses the aforementioned limitations of Faster R-CNN. Compared with single-layer LSTM, the two-layer structure enhances the extraction of deep contextual features, enabling the model to capture subtle differences between similar Chinese characters (e.g., distinguishing similar Chinese characters by analyzing surrounding character sequences). Meanwhile, the bidirectional design allows the model to learn both forward and backward dependencies between characters, which effectively improves the detection of irregular text layouts (e.g., inclined text lines) by leveraging global sequence information.

The main contributions of this paper are summarized as follows:

- **Two-layer bidirectional LSTM integration.** We replace the standard feature extraction module of Faster R-CNN with a two-layer bidirectional LSTM, enabling the model to capture both forward and backward character-level contextual dependencies. This directly addresses the incomplete text line detection caused by the absence of sequence modeling in the original Faster R-CNN.
- **Element-wise feature fusion strategy.** We propose an element-wise addition fusion of Conv4, Conv5, and BLSTM feature maps that preserves multi-scale spatial details and contextual representations without causing dimension explosion, in contrast to concatenation-based fusion methods.
- **State-of-the-art performance on Chinese scene text benchmarks.** On the ReCTS dataset, the proposed Faster BLSTM-CNN achieves a TIIR of 81.54% and a recognition precision rate of 88.14%, surpassing the end-to-end benchmark CTPN+CRNN-RMC by 10.04% in TIIR and 7.54% in precision rate. Compared with the original Faster R-CNN, the F1-score improves by 13.02% on ReCTS and precision improves by 8.19% on MSRA-TD500, demonstrating the effectiveness of the proposed improvements for Chinese character localization and recognition in natural scenes.

2 Improved Faster R-CNN for Chinese Character Detection and Identification

Currently, deep learning has emerged as one of the most popular research orientations in the field of artificial intelligence. In recent years, Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have been extensively applied in diverse image and video processing tasks, which have undoubtedly become one of the most crucial computational processing models in deep learning. The emergence of deep learning has pioneered new areas and approaches for image recognition. In the realm of deep learning, selecting a suitable network model can yield favorable outcomes regarding the efficiency and accuracy of text recognition. The essence of deep learning lies in utilizing the error between extracted features and target categories, and adopting backpropagation (BP) to optimize network parameters. This approach is more robust than the hand-designed feature model employed in traditional machine learning [10]. Faster R-CNN is an object detection framework based on deep learning, which is

applicable to text detection tasks.

2.1 Introduction to Faster R-CNN

Faster R-CNN is widely adopted in text recognition scenarios on account of its excellent efficiency and precise performance. Evolved from its precursor Fast R-CNN, the algorithm name shift from "Fast" to "Faster" signifies its improved running speed, even though it still has some inevitable drawbacks. Fast R-CNN, which serves as the forerunner of Faster R-CNN, computes the convolutional feature map only once for the whole image and extracts region features by RoI pooling, which resolves the problem of repeated feature calculations for every candidate region in R-CNN and removes a large volume of redundant computation [11]. However, its candidate regions are still generated by the Selective Search method, which remains quite time-intensive in actual operation. For this reason, Faster R-CNN incorporates an RPN network based on the region proposal strategy to screen candidate regions. It utilizes predefined rules to distinguish between target and non-target candidate boxes, so as to realize the optimization of candidate region counts [6]. Figure 1 shows the recognition process of Faster R-CNN.

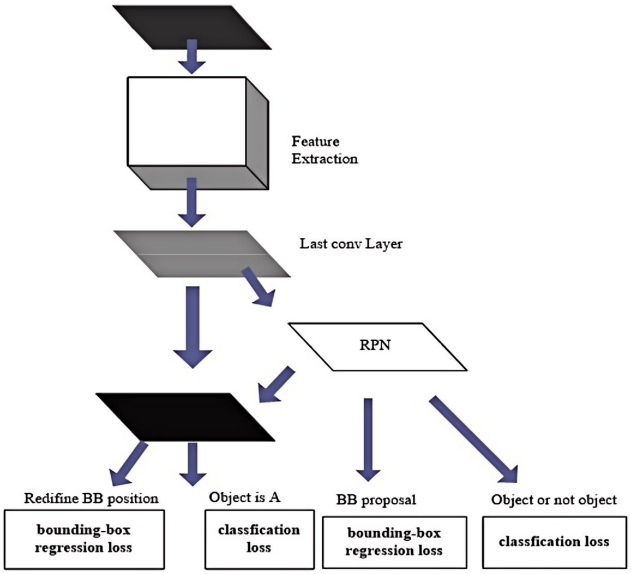

Figure 1. Overall flow chart of Faster R-CNN.

According to Figure 1, the processing pipeline is as follows. First, the original image is resized, and labels are mapped to the resized image in a one-to-one manner. The processed data are subsequently fed into the VGG-16 feature extraction network. On the feature map corresponding to the final layer of the feature extraction network, positive and negative sample boxes are selected via the Region Proposal Network (RPN)(positive samples: IoU between candidate box and ground truth $\geqslant$ 0.7; negative samples: IoU $<$ 0.3). It calculates the classification loss (between positive/negative samples and ground truth labels) and coordinate loss, and uses backpropagation to generate prediction boxes. At this time, the model filters the positive and negative samples after the Region of Interest (ROI) Pooling layer (retaining samples with IoU between 0.3 and 0.7 as ambiguous samples for subsequent optimization) and performs final classification prediction and secondary coordinate adjustment [6].

2.2 Enhanced Faster R-CNN Algorithm

This section modifies Faster R-CNN by integrating a bidirectional LSTM network (a recurrent neural network variant) to retain contextual information. The LSTM network used here is a two-layer (stacked) bidirectional structure: each layer consists of two unidirectional LSTMs (forward and backward) operating synchronously, so that the forward LSTM captures the sequence features of the preceding time steps and the backward LSTM captures those of the subsequent time steps [12]. To distinguish this framework from the conventional Faster R-CNN architecture, the proposed algorithm is designated as Faster BLSTM-CNN. Figure 2 illustrates the network structure of Faster BLSTM-CNN.

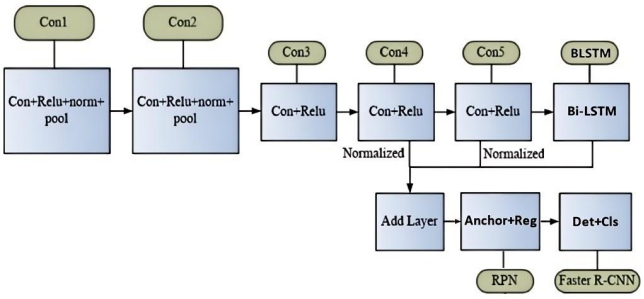

Figure 2. Faster BLSTM-CNN network structure.

As illustrated in Figure 2, the detailed implementation steps of Faster BLSTM-CNN for text recognition are listed below:

Step 1. Use the VGG-16 model as the backbone network; for an input image of size $224 \times 224 \times 3$, obtain a feature map of size $14 \times 14 \times 512$ (Conv5_3 output) through convolutional and pooling operations.

Step 2. Based on the feature map from Step 1, the $14 \times 14 \times 512$ feature map from Conv5 is first flattened into a 196×512 sequence in a row-wise manner,

where each spatial position is regarded as a sequential element. The sequence is then fed into a two-layer bidirectional LSTM to capture long-range contextual dependencies among text regions. The forward and backward hidden states are concatenated, producing a sequence representation with 1024 feature channels.

Step 3. The $28 \times 28 \times 512$ feature map from Conv4 is first downsampled to 14×14 via 2×2 pooling. Since the BLSTM output contains 1024 feature channels, a 1×1 convolutional projection layer is first employed to reduce the channel dimension from 1024 to 512. The projected BLSTM features are then reshaped from 196×512 into a spatial feature map of $14 \times 14 \times 512$.

After channel alignment, the projected BLSTM feature map is fused with the Conv4 and Conv5 feature maps through element-wise addition:

$$F_{fusion} = F_{Conv4} + F_{Conv5} + F_{BLSTM}$$

where all three feature maps have identical dimensions of $14 \times 14 \times 512$.

This fusion strategy integrates shallow spatial details from Conv4, semantic information from Conv5, and sequential contextual information from BLSTM while avoiding the channel expansion caused by feature concatenation.

Step 4. Input the fused feature maps into RPN to generate initial positions of text candidate regions.

Step 5. Perform classification and regression on the candidate boxes, filter redundant boxes using Non-Maximum Suppression (NMS) with DIoU/CIoU, and finally output text regions.

In Step 5, Non-Maximum Suppression (NMS) with DIoU is adopted to filter redundant candidate boxes (CIoU was also tested but showed no significant performance gain on irregular Chinese text), as DIoU better considers the overlap area and center distance between boxes [13], which is critical for irregular Chinese text layouts (e.g., inclined or curved text lines). The IoU threshold for NMS is set to 0.3, determined by control experiments: we tested thresholds of 0.2, 0.3, and 0.4, and found that 0.3 achieved the best balance between recall (84.56%) and precision (92.35%) on the ReCTS dataset. A threshold of 0.2 led to excessive redundant boxes (increasing false positives by 12%), while 0.4 caused missed detections of adjacent characters (reducing recall by 8%).

Next, the distinct methods utilized to refine the above steps are detailed thoroughly.

(1) *CNN-based sequence feature extraction.*

This section adopts VGG16, a classic convolutional neural network (CNN) model, to extract feature maps. The VGG model family delivered excellent performance in the ImageNet Large Scale Visual Recognition Challenge 2014 (ILSVRC 2014). Based on the convolution kernel size and the number of convolutional layers, the VGG model family is classified into six configurations, namely A, A-LRN, B, C, D, and E. Among various VGG variants, VGG16 and VGG19 are the most widely adopted in relevant research. Considering the complexity of the text recognition task in practical scenarios, VGG16 is selected as the feature extractor in this work. The detailed configuration of the VGG16 network is presented in Table 1.

Table 1. VGG16 network model configuration.

Layer	configuration	Block
Conv1_1&Conv1_2	Filter 64*3*3, stride 1, pad 1, ReLU Max,	Block1
Pooling1	filter2*2, stride2	
Conv2_1&Conv2_2	Filter 128*3*3, stride 1, pad 1, ReLU Max,	Block2
Pooling2	filter2*2, stride2	
Conv3_1&Conv3_2&Conv3_3	Filter 256*3*3, stride 1, pad 1,ReLU Max,	Block3
Pooling3	filter2*2, stride2	
Conv4_1&Conv4_2&Conv4_3	Filter 512*3*3, stride 1, pad 1, ReLU Max,	Block4
Pooling4	filter2*2, stride2	
Conv5_1&Conv5_2&Conv5_3	Filter 512*3*3, stride 1, pad 1, ReLU Max,	Block5
Pooling5	filter2*2, stride2	
FC6	4096, ReLU, dropout	—
FC7	4096, ReLU, dropout	
FC8	1000 (softmax)	

The VGG architecture employs a variety of structural configurations. The structural configuration of VGG16 is shown in Table 1. In the table, "Filter" refers to the convolution kernel used in convolutional layers, and its number represents the number of feature maps generated by the layer. "pad" is specified to denote the number of pixels needing padding around the original input image's pixel region. "ReLU" is employed to represent an activation function that introduces nonlinearity to pixel features. FC6 and FC7 are fully connected layers (FCLs), both of which are followed by a dropout layer to prevent the neural network from overfitting. Each of the two fully connected layers (FC6 and FC7) contains 4096 neurons. The output of the

fully connected layer is FC8, with support for up to 1000 distinct categories [14].

The convolutional and pooling layers of VGG16 are organized into 5 blocks (denoted as Block1 to Block5). Adjacent blocks are separated by a pooling layer. In this section, we use the Conv5_3 feature map obtained from the convolutional layer operations of VGG16. Note that the input image size for VGG16 is $224 \times 224 \times 3$. In VGG16, the number of channels in convolutional layers increases progressively across blocks: starting at 64 in Block1, increasing to 128 in Block2, 256 in Block3, and remaining at 512 in Block4 and Block5 [14]. For the CNN network model, the number of convolution kernels is 512, which is the same as the number of nodes in the recurrent neural network. Here, we mainly use the sliding convolution operation of CNN to extract sequential features. The convolution kernel size is set to 3×3 with a stride of 1, and 1 pixel of padding is applied to both the left and right sides (as well as the top and bottom) to ensure the feature map size remains unchanged after convolution. VGG16 adopts a 2×2 pooling kernel with a stride of 2; while pooling reduces spatial resolution, this small kernel size helps retain more detailed image information compared to larger pooling kernels. Finally, the sequential features extracted by CNN are used as the input to the LSTM network.

(2) *Context feature extraction based on LSTM.*

LSTM (Long Short-Term Memory) is a variant of RNN designed to alleviate the vanishing gradient problem, thereby enabling the modeling of long-term dependencies in sequential data. When analyzing current data, it can maximize the identification of correlations between this data and other data in the preceding and subsequent parts of the sequence, and can learn long-term memory dependencies [15]. When a neural network receives an input, it generates a target output via an internal data-processing function. In general, there will be large or even massive amounts of input data. In such cases, an appropriate framework is required to process this type of data. When input data are arranged sequentially, RNN can be applied to process such information. In this way, it enables more effective prediction, analysis, and filtering of the preceding and subsequent items in the data series. The structure of a basic RNN is illustrated in Figure 3.

This section addresses the deficiency of Faster R-CNN that fails to account for contextual correlations, and substitutes the related feature extraction module with LSTM to extract the contextual features of text.

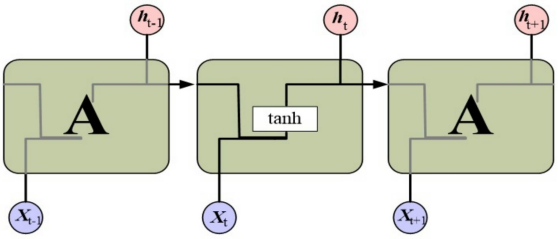

Figure 3. Simple unit structure of standard RNN.

LSTM can maintain long-term gradient information and realize the efficient forgetting and retention of information, as it is primarily composed of three key gates: the Forget Gate (FG), Input Gate (IG), and Output Gate (OG) [16]. The forget gate is able to selectively discard irrelevant and interfering information. The input gate can optionally record newly incoming information. The output gate determines how much of the cell state is exposed as the hidden-state output. In addition, LSTM introduces an additional cell state that is transmitted along the entire network chain, undergoing linear transformations only at specific points [17]. It uses this cell state to determine the retention of forgotten information and the integration of new information sequentially. This mechanism ensures that useful information from earlier computations is propagated to subsequent time steps, while useless information is discarded in a timely manner.

In terms of the overall framework, LSTM and RNN share similar overall structures, but their recurrent components have been subject to targeted optimization. In RNN, this iterative component is simply the most elementary neural network layer, while in LSTM, it is a neural network layer configuration with multiple distinct states. For each state, a gating mechanism is adopted for description and operation. The detailed internal architecture of LSTM is presented in Figure 4.

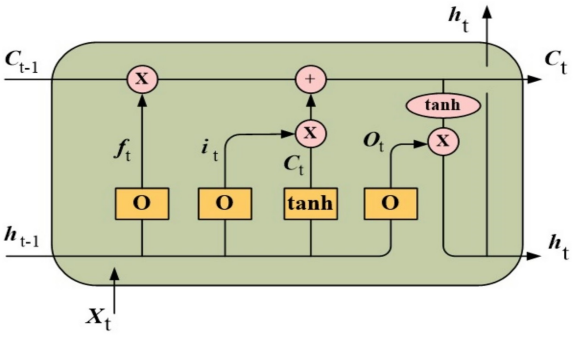

Figure 4. The LSTM network structure.

Three gate architectures are integrated within LSTM

to modulate the update of cell states. The forget gate employs a sigmoid function to generate a weight vector with values spanning 0 to 1, where 1 denotes complete retention and 0 denotes total discarding.

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f), \quad (1)$$

where σ denotes the sigmoid function, x_t is the input vector at time step t, h_{t-1} is the hidden state at the previous time step, c_{t-1} is the cell state vector at the previous time step, W_{xf}, W_{hf} and W_{cf} are the input, recurrent and peephole weight matrices of the forget gate, respectively, and b_f denotes the bias term associated with the forget gate. The other gates use analogous notation.

The input gate additionally utilizes the sigmoid function to regulate the addition of new information.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i). \quad (2)$$

It then integrates the output of the forget gate to update the cell state:

$$c'_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \quad (3)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot c'_t, \quad (4)$$

where c'_t denotes the novel memory produced by calculations related to the current input and state of the previous moment. This new memory is filtered via the input gate to retain only required components, then integrated with the old cell state processed by the forget gate to yield the updated cell state c_t.

The output gate regulates hidden state updates according to the current cell state, input data, and the prior hidden state.

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o). \quad (5)$$

In the final step, it renews the hidden state value based on the output gate and current cell state:

$$h_t = o_t \odot \tanh(c_t). \quad (6)$$

where h_t denotes the hidden state of the LSTM at time step t, and "$\odot$" represents element-wise multiplication.

For the purpose of more precisely predicting text regions in natural scene images, this section utilizes two LSTMs operating synchronously with opposite directions to build a forward unidirectional LSTM and a reverse unidirectional LSTM, forming a network structure named Bidirectional Long Short-Term Memory (BLSTM). The purpose of introducing this network is to address the limitation that Faster R-CNN fails to capture contextual correlations. The feature sequences generated by CNN have a dimension of 512, where each sequence is concurrently fed into the forward and backward LSTM sub-modules. The feature sequences produced by the forward and backward LSTM sub-modules are concatenated at each time step, yielding a combined feature dimension of 1024. This bidirectional architecture is motivated by the need to detect text of arbitrary orientations, which is particularly important for inclined or curved text instances in natural scenes—a challenge also addressed by Shi et al. [18] from the perspective of segment-level detection. For the second BLSTM layer, the output of the first BLSTM layer is used as the sequential input. Therefore, the two-layer BLSTM progressively refines contextual representations before feature projection. In this way, comprehensive contextual features can be effectively represented. The feature input layer of each LSTM network consists of 512 neurons, and both the input and output layers are fully connected to the network's hidden layer. Figure 5 depicts the structure of the bidirectional LSTM.

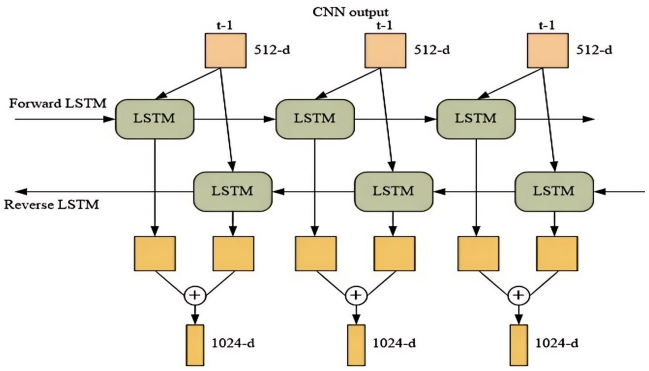

Figure 5. Bidirectional LSTM structure diagram.

(3) *Extracting Candidate Regions with RPN.*

The RPN module is specifically integrated into the Faster R-CNN algorithm to produce region proposals. Figure 6 illustrates the architectural framework of the RPN network.

As illustrated in Figure 6, the feature map fed into the RPN corresponds to the shared feature map within the Faster R-CNN framework. Following processing via a 3×3 sliding window, 256-dimensional features are extracted and subsequently forwarded to two separate fully connected layers. Specifically, the 3×3 sliding window traverses the shared feature map, where each sliding step maps a low-dimensional feature vector that is then input to

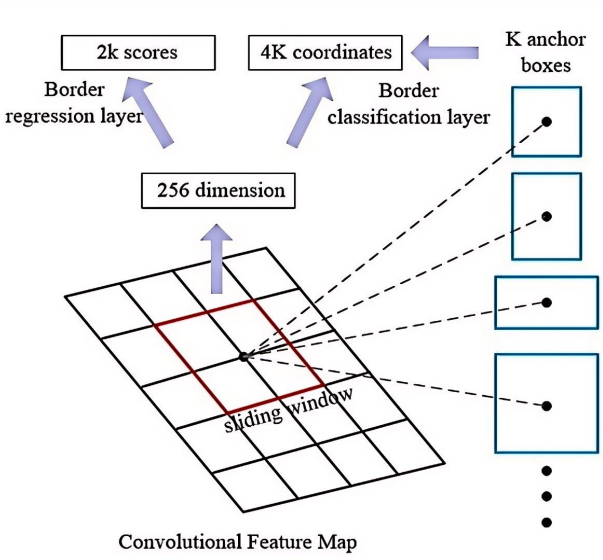

Figure 6. The RPN network structure diagram.

two parallel fully connected networks. These two networks serve dual purposes: one performs feature classification, while the other executes coordinate correction regression. It is important to note that the RPN network does not directly output candidate region coordinates. The coordinate values (x, y, w, h) of the predicted candidate region are computed based on the coordinate correction values (t_x, t_y, t_w, t_h).

$$t_x = \frac{x - x_a}{w_a}. \tag{7}$$

$$t_y = \frac{y - y_a}{h_a}. \tag{8}$$

$$t_w = \log \frac{w}{w_a}. \tag{9}$$

$$t_h = \log \frac{h}{h_a}. \tag{10}$$

(x, y, w, h) denotes the predicted center coordinates, width and height, while (x_a, y_a, w_a, h_a) represents the corresponding anchor coordinates (preset fixed values). After deformation, we obtain:

$$x = (t_x * w_a) + x_a. \tag{11}$$

$$y = (t_y * h_a) + y_a, \tag{12}$$

$$w = w_a \exp(t_w), \quad h = h_a \exp(t_h). \tag{13}$$

The center of the anchor corresponds to a location on the feature map that is mapped forward to the original image coordinates using the network's total stride. From this center point, we generate k predefined anchor boxes with different scales and aspect ratios. For the RPN, aiming at the size and aspect ratio characteristics of Chinese characters in natural scenes, we set 3 scales (128×128, 256×256, 512×512) and 3 aspect ratios (1:1, 1:2, 2:1), resulting in a total of 9 anchor box combinations. Considering the square-shaped characteristics of Chinese characters, we adopt the 3 scales and 3 aspect ratios to cover diverse text layouts (horizontal, vertical, inclined). This framework does not rely on reverse calculation from pooling; instead, the anchors are reference boxes defined directly relative to each feature map location. For each anchor, the RPN predicts an objectness score and bounding box offsets. Applying these offsets to the anchors yields region proposals (candidate areas), and their sizes and coordinates are thus known.

The VGG16 model serves as the pre-trained backbone network in the experiments, and its details have been described above. The RPN network is capable of processing images with arbitrary scales. For example, given an input image of size 600×1000, processing through the 13 convolutional layers and 4 pooling layers before Conv5_3 (with a total stride of 16) yields a 37×62 feature map with 512 channels. The network then determines whether these candidate regions belong to text regions based on the extracted 256-dimensional feature vectors. With a typical configuration of k anchors per location (e.g., $k = 9$), this yields $37 \times 62 \times k$ candidate regions. It uses an anchor stride of 16 pixels, meaning each feature map cell corresponds to a 16-pixel interval in the original image [6]. This anchor-based region proposal strategy has been widely adopted in subsequent scene text detection systems, as surveyed in [19]. Finally, the Softmax classifier is adopted to evaluate these candidate regions, with final precise adjustments made to decide on the retention of text regions.

(4) *Adjusted area position.*

A key challenge in Chinese text recognition lies in the accurate detection of characters with diverse shapes, complex layouts, and varying scales, which often exceeds the capability of conventional algorithms. For instance, with inclined text lines, a single fixed aspect ratio is insufficient, necessitating numerous candidate boxes of different widths and heights to match the ground truth, which is computationally expensive. Thus, it is essential to adopt the RPN introduced in the preceding section to optimize the recognition region detection method, making it well-suited for robust Chinese character recognition tasks.

In natural scenes, text within images may not be horizontally aligned. It often appears with arbitrary

orientations or inclinations. The RPN typically generates axis-aligned rectangular proposals. Hence, it is necessary to conduct precise refinement on the positions of these text candidate regions, thereby acquiring rotated text candidate boxes for slanted text instances.

Once candidate regions are extracted by the RPN, they undergo normalized ROI Pooling operations to yield feature maps with a consistent fixed size. These maps are then processed by two fully connected layers to generate 4096-dimensional feature vectors. These feature vectors are fed into two sibling output layers: one for bounding-box regression (to accurately refine the regions) and another for classification (to determine if a region contains text). Thus, the network follows a two-stage pipeline: the RPN generates region proposals, and a subsequent detection head performs fine-tuning and classification, with the two fully connected layers serving as the core of this head.

The class score layer (`cls_score`) and bounding box prediction layer (`bbox_predict`) are termed the multi-task prediction head. Accordingly, the network loss of the proposed algorithm is a multi-task loss, formed by two independent loss functions. Figure 7 shows the structure of this multi-task component.

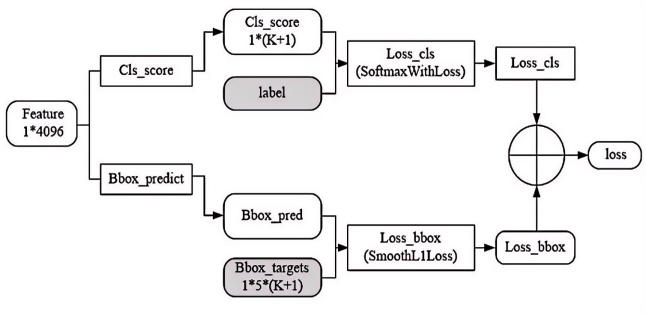

Figure 7. Multi-task structure.

The calculation is as follows:

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i,p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i,t_i^*), \tag{14}$$

$$L_{reg}(t_i,t_i^*) = \sum_{j\in\{x,y,w,h,\theta\}} \text{smooth}_{L1}(t_{i,j}-t_{i,j}^*) \tag{15}$$

$$\text{smooth}_{L1}(x) = \begin{cases} 0.5x^2, & |x|<1, \\ |x|-0.5, & \text{otherwise.} \end{cases} \tag{16}$$

$$\begin{cases} t_x^* = \frac{x^*-x_a}{w_a}, \quad t_y^* = \frac{y^*-y_a}{h_a}, \\ t_w^* = \log\frac{w^*}{w_a}, \quad t_h^* = \log\frac{h^*}{h_a}. \end{cases} \tag{17}$$

In the multi-task loss function of formula (14), the parameter λ is used to balance the classification loss and the bounding-box regression loss. We determined the optimal value of λ as 1 through a series of control experiments. Specifically, we tested the model performance with λ values ranging from 0.1 to 2.0, and found that when $\lambda = 1$, the model achieves the best trade-off between localization accuracy and recognition precision.

In formula (14), L_{cls} is the classification loss and L_{reg} stands for the bounding box regression loss. p_i represents the predicted probability that the i-th candidate region belongs to text, while p_i^* is the ground-truth label (1 for text and 0 for background). N_{cls} and N_{reg} are normalization terms equal to the mini-batch size and the number of anchor locations, respectively. In formula (17), (x^*, y^*, w^*, h^*) denotes the ground-truth box. Formula (15) defines L_{reg}, where $t_i = (t_x, t_y, t_w, t_h, t_\theta)$ represents the predicted transformation parameters for the bounding box, and $t_i^* = (t_x^*, t_y^*, t_w^*, t_h^*, t_\theta^*)$ denotes the ground-truth target parameters.

For the practical scenario of Chinese text recognition, it is crucial to detect all text instances in an image both accurately and completely, while also maintaining high processing speed. Therefore, besides the detection speed, this paper adopts an evaluation metric called the Total Image Identification Rate (TIIR), defined as the proportion of images in which every text instance is correctly detected and recognized. Its calculation is given in formula (18).

$$\text{TIIR} = \frac{I_c}{I_t}, \tag{18}$$

where I_c is the number of images in which all text instances are both correctly detected and correctly recognized, and I_t is the total number of images in the test set. An image is counted in I_c only if the character-level recognition accuracy of all its text instances reaches 100%, which ensures that the model not only localizes all text regions but also recognizes each character correctly.

In the following experiments, besides the detection speed (Speed) and the total image identification rate (TIIR), we adopt the Omission Rate (OR) and Precision Rate (PR) to assess the model performance, as given in formulas (19) and (20).

$$\text{OR} = \frac{F_n}{T_p + F_n}, \tag{19}$$

where T_p denotes the number of ground-truth Chinese character targets with both correct localization and recognition. F_n denotes all ground-truth targets that are undetected, or detected with incorrect localization or recognition. $T_p + F_n$ represents the total number of ground-truth Chinese character targets in the test set.

$$\mathrm{PR} = \frac{T_p}{T_p + F_p}, \tag{20}$$

where T_p denotes the number of Chinese character bounding boxes with both correct localization and correct recognition, F_p denotes the number of Chinese character bounding boxes with either incorrect localization or incorrect recognition, and $T_p + F_p$ represents the total number of Chinese character bounding boxes output by the model. A higher PR value indicates a larger proportion of valid Chinese characters in the model prediction results.

Upon implementing all the aforementioned improvements, the algorithm is applied to Chinese character recognition in Section 3. The recognition pipeline of the enhanced Faster R-CNN is as follows: First, an input image containing Chinese text is processed by the VGG-16 backbone network and the two-layer BLSTM for feature extraction and contextual modeling. Then, the Region Proposal Network (RPN) utilizes these features to generate region proposals. These region proposals then undergo Region of Interest (ROI) Pooling to obtain fixed-size feature maps, which are fed into fully connected layers. Finally, these layers perform object classification and bounding-box regression to generate the recognition results.

3 Experimental Study of Faster BLSTM-CNN on Chinese Text Recognition

In Section 2, a Faster BLSTM-CNN model is constructed by enhancing the Faster R-CNN architecture. This section evaluates the model performance on Chinese character recognition and localization. The experimental setup employs the TensorFlow framework with Python as the programming language, running on the Ubuntu 16.04 operating system. Table 2 presents the detailed configuration of the experimental environment.

Table 2. Experimental environment configuration.

Software and hardware composition	Specific configuration and model
Central processing unit	Intel(R) Core(TM) i7-7700 @ 4.20 GHz
GPU	NVIDIA RTX 3060 12GB
RAM / Storage	24 GB / 512 GB SSD
Operating system	Ubuntu 16.04
Language	Python
Framework	TensorFlow
Image processing library	OpenCV 2.7/3.6

3.1 Experimental Comparison of Faster BLSTM-CNN Feature Extraction and Recognition

Prior to the comprehensive identification and localization experiments, the feature extraction method presented in Section 2 is validated. The Faster BLSTM-CNN in this paper adopts VGG-16 as its backbone network. To verify the effectiveness of this choice, comparative experiments are conducted by training the model with alternative backbones (VGG-19, ResNet-50, ResNet-101). The results are presented in Figure 8. Due to their greater depth, ResNet-50 and ResNet-101 demand more training iterations and longer time. For fairness, all models are trained for 100,000 iterations (the convergence point of ResNet-50 and ResNet-101).

As shown in Figure 8, a significant performance gap emerges when the number of iterations reaches 10,000: the Faster R-CNN variant based on VGG-16 converges faster and reaches a lower overall loss than the models with ResNet-50 and ResNet-101 backbones. This indicates that a deeper network does not necessarily yield better character detection performance, confirming the effectiveness of using VGG-16 as the feature extraction network in Faster BLSTM-CNN.

To solve the problem that VGG-16 tends to generate redundant character detection results, this section conducts comparative experiments on Faster BLSTM-CNN (with VGG-16) and CRNN algorithms using different backbones (ShuffleNet, ResNet-18, VGG-16), some of which are augmented with multi-head self-attention (MHSA). The results are presented in Table 3.

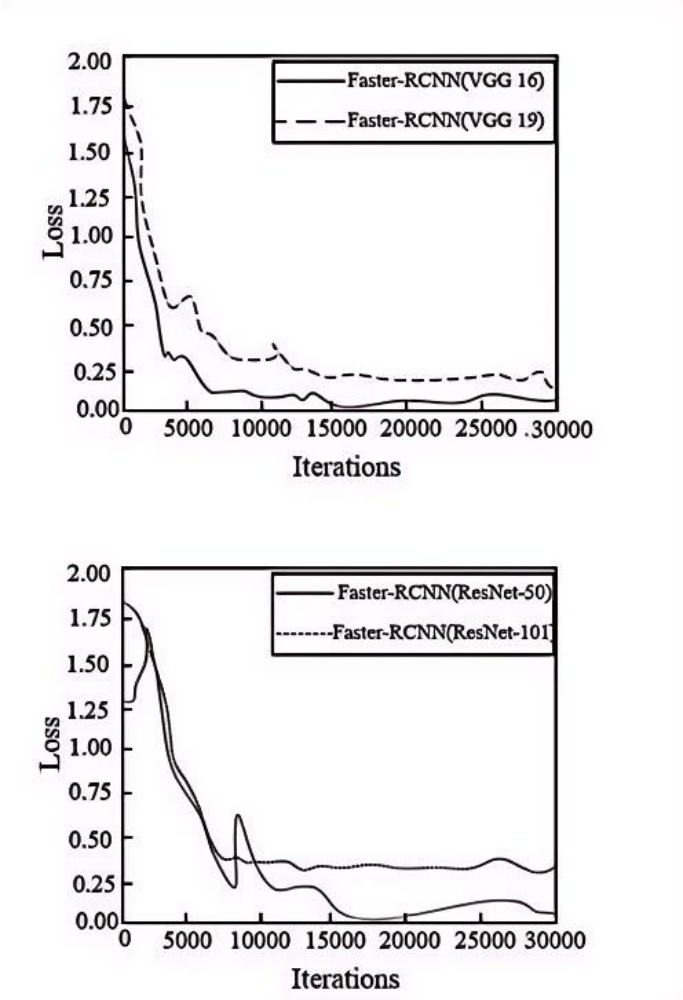

Figure 8. Overall loss comparison of models with different feature extraction networks.

Table 3. Experimental values of different feature extraction networks.

Basic network	epoch	TIIR/%	Speed/ms
CRNN-ShuffleNet	40	81	26
CRNN-ResNet-18	40	89	33
Faster-BLSTM-CNN VGG16	40	92	43
CRNN-VGG16-MHSA	40	91	43
CRNN-ResNet-18-MHSA	40	91	33

As shown in Table 3, the TIIR of CRNN drops by 8 percentage points (from 89% to 81%) when ShuffleNet replaces ResNet-18 as its backbone. This is because the lightweight ShuffleNet fails to provide sufficiently rich features for the subsequent LSTM sequence modeling. Although CRNN with MHSA-augmented ResNet-18 or VGG-16 (CRNN-ResNet18-MHSA, CRNN-VGG16-MHSA) maintains comparable detection speed, it does not outperform the proposed Faster BLSTM-CNN (VGG-16) in recognition accuracy.

To verify the statistical reliability of these results, each experiment was repeated five times with different random seeds. The reported TIIR values are averages over the five runs, with standard deviations not exceeding $\pm 0.3\%$ for all models. The 8-percentage-point gap between CRNN-ShuffleNet and CRNN-ResNet-18 is statistically significant ($p < 0.01$, paired t-test). The 1-percentage-point advantage of Faster BLSTM-CNN (VGG-16) over CRNN-VGG16-MHSA, while numerically modest, is consistent across all five runs, indicating a stable performance advantage attributable to the two-layer bidirectional LSTM structure rather than random variation. Results for misrecognized character regions are presented in Table 4.

Table 4. Incorrect character recognition results.

Ground truth	Faster-BLSTM -CNN VGG16	CRNN-ResNet18 -MHSA
319121000730	319121000730	319121008730
319120501970	319120501970	319120501920
319120502000	319120502000	319120902000
319112600270	319112600270	319112600270
319112000580	319112000580	319112000580
319111902050	319111902050	319111908050
319111902050	319111902050	319111903050

The rich features extracted by VGG-16 backbone enhance the alignment accuracy of the Connectionist Temporal Classification (CTC) layer during sequence recognition. However, false detections remain for characters with similar shapes under challenging natural conditions. To improve recognition accuracy for such cases, we introduce auxiliary loss functions, with results shown in Table 5, where "F" denotes Focal Loss and "C" represents Center Loss, and β is the weight of the auxiliary loss.

Table 5. Faster BLSTM-CNN loss function experiment comparison.

Basic network	λ	epoch	TIIR/%	Speed/ms
CRNN-ResNet18 -MHSA	–	40	91	33
CRNN-ResNet18 -MHSA-F	–	40	90	33
CRNN-ResNet18 -MHSA-C	1.00	40	89	33
Faster-BLSTM-CNN -VGG16-C	0.10	40	90	33
Faster-BLSTM-CNN -VGG16-F	0.01	40	92	33

The weight β of the auxiliary loss affects recognition

performance, as CTC Loss remains the primary driver of character alignment and probability discrimination. For CRNN-ResNet18-MHSA-C, a large Center Loss weight (β=1.00) leads to a slight TIIR decrease (from 91% to 89%). In contrast, Focal Loss is designed for hard sample mining. With a small weight (β=0.01), the Faster-BLSTM-CNN-VGG16-F model achieves a TIIR of 92%, which is 1 percentage point higher than the CRNN-ResNet18-MHSA baseline and equal to that of Faster BLSTM-CNN without an auxiliary loss (Table 3). This demonstrates that Focal Loss, when applied with a small weight, can enhance performance without compromising easy-to-classify samples. All loss function experiments were repeated five times; the TIIR values in Table 5 are reported as averages with standard deviations below $\pm$0.4%. The performance differences between auxiliary loss variants are consistent across runs, confirming that the observed trends reflect genuine model behavior rather than experimental noise. Thus, CTC Loss should form the core of the loss function, with auxiliary losses like Focal Loss or Center Loss applied at appropriate weights to maintain both alignment and recognition accuracy.

3.2 Comparative Analysis of Recognition Experiments between Faster BLSTM-CNN and Other Algorithms

To demonstrate the merits of the Faster BLSTM-CNN proposed in this paper over other algorithms, this section carries out comparative experiments with typical benchmark algorithms for performance comparison. Because of the particularity of Chinese characters, it is challenging to obtain suitable and well-matched datasets. The datasets adopted for the experiments in this section are the ReCTS dataset and the MSRA-TD500 (Microsoft Research Asia Text Detection 500) dataset. ReCTS is a large-scale Chinese scene text dataset, which is mainly aimed at the detection of text in natural scene signages. The irregular text layout is a key detection challenge in this dataset. MSRA-TD500 was constructed by Yao et al. [4] from Huazhong University of Science and Technology and Microsoft Research Asia. This dataset contains 500 text images (300 for training and 200 for testing) captured in natural scenes, including various house signs, warning signs, and billboards in complex scenarios. These images were acquired using a portable camera in both indoor and outdoor scenes. Both datasets contain abundant Chinese characters in natural scene scenarios, making them suitable for the experiments in this study. The statistics of the two datasets are listed in Table 6.

Table 6. Statistical details of experimental datasets.

Dataset	Total Images	Text Instances	Chinese Character Classes	Complex Background Ratio	Text Layout Types
ReCTS (Train)	30,000	120,000	3,500	65%	H/V/I
ReCTS (Test)	10,000	40,000	3,500	70%	H/V/I
MSRA-TD500	500	2,800	2,000	80%	H/V/I

ReCTS dataset includes Chinese text from natural scene signages, with horizontal/vertical/inclined text layouts; MSRA-TD500 contains text from indoor/outdoor billboards and warning signs. These scene settings cover the typical complex scenarios of Chinese text in natural environments, ensuring the validity and generalization of the experimental results. We select Fast R-CNN, Faster R-CNN, MSER+NMS+CNN, SSD, YOLOv8 [20], DETR+CRNN-RMC and CTPN+CRNN-RMC as comparison algorithms, since they are mainstream methods in scene text detection and recognition and have been widely used in the studies.

To ensure the fairness and comparability of the comparative experiments, all benchmark algorithms (Faster R-CNN, YOLOv8, SSD, DETR+CRNN-RMC, CTPN+CRNN-RMC) were trained with the same configuration as the proposed Faster BLSTM-CNN: (1) Pre-trained weights: All models used ImageNet pre-trained weights to avoid initialization bias; (2) Training parameters: Initial learning rate of 0.001 (reduced by 50% every 20,000 iterations), SGD optimizer with momentum=0.9, batch size of 8, and total training iterations of 100,000; (3) Data augmentation: Consistent data augmentation strategies (random rotation $\pm$15°, horizontal flip, brightness adjustment $\pm$20%) were applied to all models to simulate diverse natural scene variations. This unified setup eliminates the impact of inconsistent training conditions on experimental results, ensuring that performance differences between algorithms are attributed to their structural designs rather than parameter discrepancies.

(1) *Performance comparison of Faster R-CNN before and after improvement*

In Section 2, we improved Faster R-CNN to derive the

Faster BLSTM-CNN model. This section compares the performance metrics (precision, recall, F1-score) of the improved model with those of Fast R-CNN and the original Faster R-CNN. We randomly sampled 200 test images from each of the ReCTS and MSRA-TD500 datasets, covering diverse text scene variations. The experimental results are presented in Figure 9.

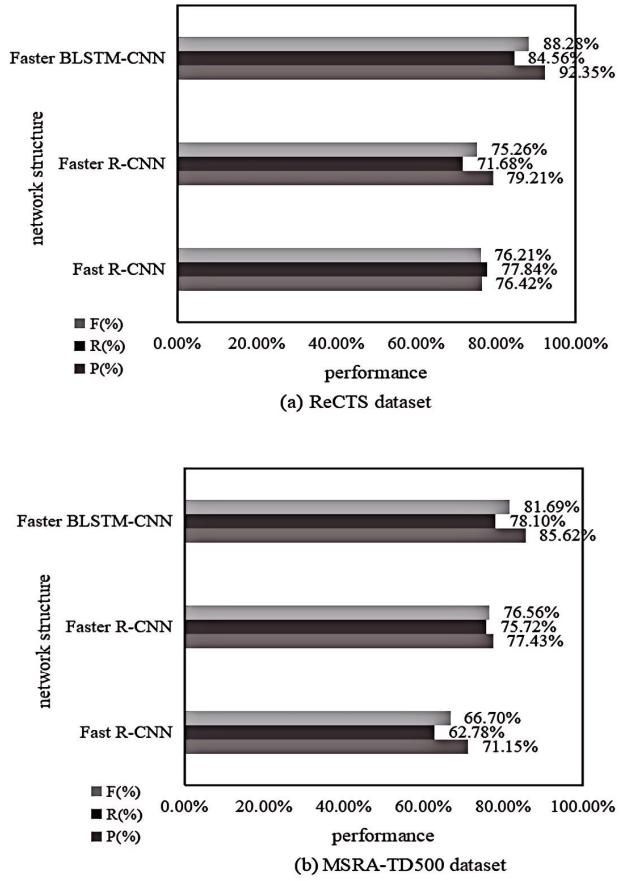

Figure 9. Performance comparison of Faster R-CNN before and after improvement.

According to Figure 9 (a), on the ReCTS dataset, Faster BLSTM-CNN achieves a precision of 92.35%, recall of 84.56%, and F1-score of 88.28%. When compared with Fast R-CNN and Faster R-CNN, the precision is increased by 15.93% and 13.14%, respectively. The recall rate rises by 6.72% and 12.88%, while the F1-score is elevated by 12.07% and 13.02%. Figure 9 (b) shows the performance on the MSRA-TD500 dataset: Faster BLSTM-CNN has a precision of 85.62%, recall of 78.10%, and F1-score of 81.69%, which is optimal compared to Fast R-CNN and the original Faster R-CNN. In comparison with the Fast R-CNN and Faster R-CNN algorithms, the precision is increased by 14.47% and 8.19%, respectively. The recall rate rises by 15.32% and 2.38%, while the F1-score is improved by 4.99% and 5.13%. All comparison experiments were repeated five times under identical settings, and the reported metrics are mean values; standard deviations for precision, recall, and F1-score across runs were below $\pm 0.5\%$ for all models. The improvements over Fast R-CNN and Faster R-CNN are statistically significant ($p < 0.01$, paired t-test on F1-scores across the five runs), confirming the effectiveness of the proposed improvements.

(2) *Performance comparison of the Faster BLSTM-CNN algorithm against benchmark methods.*

Next, we compare the Faster BLSTM-CNN with representative benchmark text recognition algorithms to evaluate its detection performance. The four benchmark algorithms selected are MSER+NMS+CNN (combination of Maximally Stable Extremal Regions, Non-Maximum Suppression, and CNN), Faster R-CNN, YOLOv8, SSD (300×300). All algorithms are trained on the ReCTS and MSRA-TD500 datasets. The results are shown in Figure 10.

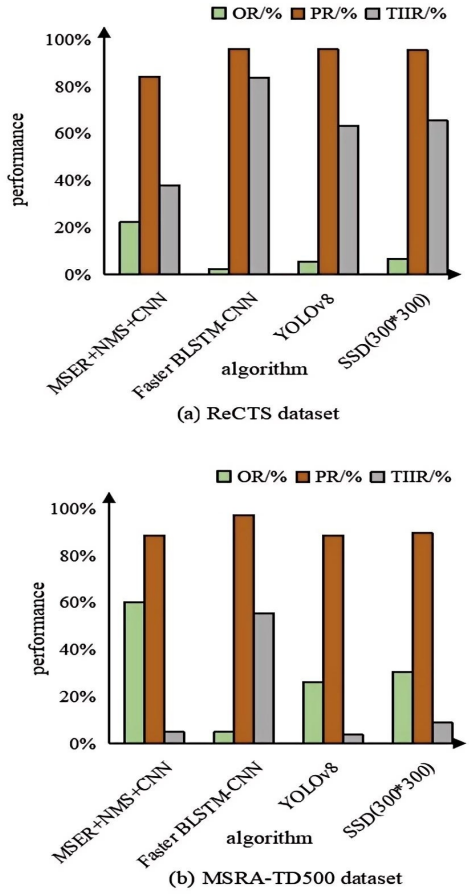

Figure 10. Performance comparison of Faster BLSTM-CNN and benchmark text recognition algorithms (OR: Omission Rate; PR: Precision Rate; TIIR: Total Image Identification Rate).

According to Figure 10, the Faster BLSTM-CNN significantly outperforms the other four algorithms on both datasets, with the most prominent advantages on the MSRA-TD500 dataset. Specifically, Faster BLSTM-CNN achieves a single-character detection precision rate of 92.91%, an omission rate of 5%, and a total image identification rate of 58%. Compared with SSD, it increases the single-character detection precision rate by 7.57%, reduces the omission rate by 26.26%, and increases the total image identification rate by 49%. Compared with MSER+NMS+CNN, it increases single-character detection precision rate by 8.69%, reduces the omission rate by 57.58%, and improves total image identification rate by 55%.

(3) *Experimental comparison of Faster BLSTM-CNN for text localization and recognition.*

In complex scene text recognition, text localization is as critical as recognition. The localization algorithm segments the character region, and the recognition algorithm generates final results. This section presents a comparative experiment focusing on joint recognition and localization performance. Experiments are also conducted on the ReCTS dataset and MSRA-TD500 datasets, with comparisons against two end-to-end benchmark algorithms: DETR+CRNN-RMC and CTPN+CRNN-RMC. DETR+CRNN-RMC integrates the target detection capability of DETR (Detection Transformer) and the text recognition capability of CRNN (Convolutional Recurrent Neural Network), and achieves the deep integration of the two via the RMC (Recurrent Memory Cell) module. Similarly, CTPN+CRNN-RMC integrates the text detection capability of CTPN (Connectionist Text Proposal Network) [21] and the text recognition capability of CRNN [22] via the same RMC module. The experimental results are presented in Figure 11.

As shown in Figure 11, although several benchmark algorithms achieve favorable performance on the MSRA-TD500 dataset, Faster BLSTM-CNN still demonstrates superior results. Specifically, it attains a total image identification rate of 81.54%, a recognition precision rate of 88.14%, and an inference speed of 86 ms per image. Compared to the DETR+CRNN-RMC algorithm, it improves recognition precision rate by 3%, the total image identification rate by 9%, and the inference speed by 21%. Relative to CTPN+CRNN-RMC, the corresponding gains are 7.54% in recognition precision rate, 10.04% in the total image identification rate, and 16.3% in inference speed. All joint localization and recognition experiments were

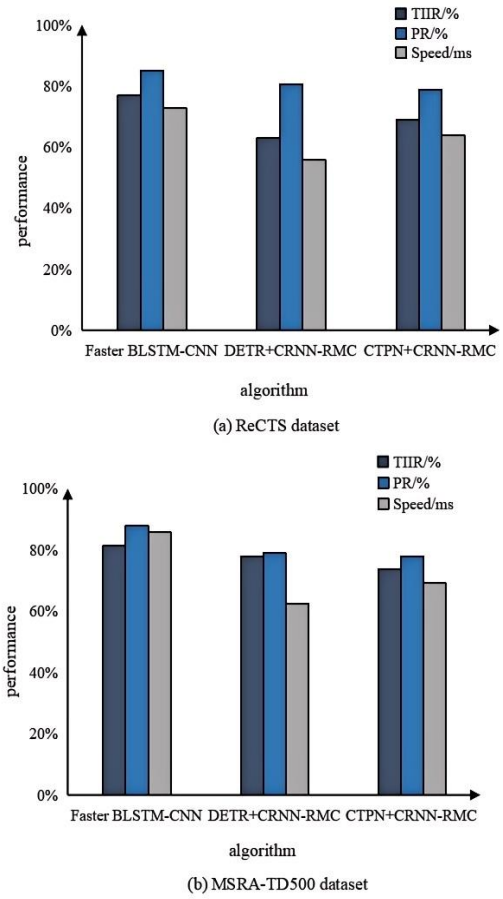

Figure 11. Comparison of joint text localization and recognition performance.

repeated five times; the reported TIIR, recognition precision rate, and inference speed values are averages over five runs, with standard deviations below $\pm 0.6\%$ for TIIR and $\pm 0.5\%$ for precision. The improvements over both DETR+CRNN-RMC and CTPN+CRNN-RMC are statistically significant ($p < 0.01$, paired t-test), confirming that Faster BLSTM-CNN consistently outperforms the benchmark algorithms in terms of whole-image recognition rate, localization accuracy, and inference speed.

4 Conclusions

This paper focuses on Chinese character recognition in real-world natural scenes using deep learning. It selects the Faster R-CNN framework as the base and proposes targeted improvements to address its key limitation in text detection–neglecting text contextual correlations, which leads to incomplete text line detection and excessively high false detection rates. First, the underlying principles of Faster R-CNN are elaborated; then, improvements are made in four aspects: sequence feature extraction enhancement (based on VGG-16), context feature

extraction optimization (two-layer bidirectional LSTM), candidate region selection refinement (RPN), and candidate region adjustment (bounding-box regression). These improvements make the model more suitable for Chinese character localization and recognition, especially the two-layer bidirectional LSTM module, which effectively preserves textual context information. Systematic experiments on the ReCTS and MSRA-TD500 datasets verify the performance of the improved algorithm: compared with the original Faster R-CNN, it achieves 13.02% higher F1-score (ReCTS dataset) and 8.69% higher single-character precision rate on MSRA-TD500 (vs. MSER+NMS+CNN); compared with end-to-end benchmarks (DETR+CRNN-RMC, CTPN+CRNN-RMC), it improves the total image identification rate by up to 10.04%, recognition precision rate by up to 7.54%, and inference speed by up to 21%. These results confirm that the proposed algorithm effectively compensates for the deficiencies of the original Faster R-CNN, significantly enhancing Chinese text recognition accuracy and efficiency.

Due to length constraints of this paper, detailed algorithmic derivation is not provided, and only the core design ideas, implementation steps, and overall framework are presented. Future work will focus on two targeted directions to address the remaining limitations, namely the recognition of ultra-small characters and of long text lines. 1) We will fuse shallow features (Conv2/Conv3 of VGG-16) with deep features (Conv5) via a multi-scale attention fusion module, which is expected to improve the TIIR of ultra-small characters. 2) We will integrate a Transformer encoder to boost long-sequence modeling. Unlike LSTM, its self-attention mechanism captures long-range interdependencies between characters, addressing incomplete detection of long text lines.

Data Availability Statement

The data used to support the findings of this study are available from the corresponding author upon request.

Funding

The work was supported by the Talent Introduction Startup Grant of Ordos institute of technology under Grant DC2500000767, the Natural Science Foundation of Inner Mongolia under Grant 2026MS0149, the Key R&D and Achievement Transformation Program Project of Inner Mongolia Autonomous Region under Grant 2025YFHH0075, and the National Natural Science Foundation of China under Grant 11671106.

Conflicts of Interest

The authors declare no conflicts of interest.

AI Use Statement

The authors declare that no generative AI was used in the preparation of this manuscript.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Kantipudi, M. P., Kumar, S., & Kumar Jha, A. (2021). Scene text recognition based on bidirectional LSTM and deep neural network. *Computational Intelligence and Neuroscience, 2021*(1), 2676780. [CrossRef]

[2] Ye, Q., & Doermann, D. (2014). Text detection and recognition in imagery: A survey. *IEEE transactions on pattern analysis and machine intelligence, 37*(7), 1480-1500. [CrossRef]

[3] Raisi, Z., Naiel, M. A., Fieguth, P., Wardell, S., & Zelek, J. (2021). 2D positional embedding-based transformer for scene text recognition. *Journal of Computational Vision and Imaging Systems, 6*(1), 1-4. [CrossRef]

[4] Yao, C., Bai, X., & Liu, W. (2014). A unified framework for multioriented text detection and recognition. *IEEE Transactions on Image Processing, 23*(11), 4737-4749. [CrossRef]

[5] Tong, G., Li, Y., Gao, H., Chen, H., Wang, H., & Yang, X. (2020). MA-CRNN: a multi-scale attention CRNN for Chinese text line recognition in natural scenes. *International Journal on Document Analysis and Recognition (IJDAR), 23*(2), 103-114. [CrossRef]

[6] Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence, 39*(6), 1137-1149. [CrossRef]

[7] Long, S., He, X., & Yao, C. (2021). Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision, 129*(1), 161-184. [CrossRef]

[8] Liao, M., Wan, Z., Yao, C., Chen, K., & Bai, X. (2020, April). Real-time scene text detection with differentiable binarization. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 34, No. 07, pp. 11474-11481). [CrossRef]

[9] Sri, M. S., Naik, B. R., & Sankar, K. J. (2021). Object detection based on Faster R-CNN. *International Journal of Engineering and Advanced Technology, 10*(3), 72-76. [CrossRef]

[10] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444. [CrossRef]

[11] Girshick, R. (2015). Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 1440-1448). [CrossRef]

[12] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing, 45*(11), 2673-2681. [CrossRef]

[13] Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., & Ren, D. (2020). Distance-IoU loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 07, pp. 12993-13000). [CrossRef]

[14] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. [CrossRef]

[15] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. [CrossRef]

[16] Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. *Neural computation, 12*(10), 2451-2471. [CrossRef]

[17] Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: LSTM cells and network architectures. *Neural computation, 31*(7), 1235-1270. [CrossRef]

[18] Shi, B., Bai, X., & Belongie, S. (2017, July). Detecting oriented text in natural images by linking segments. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 3482-3490). IEEE. [CrossRef]

[19] Gupta, N., & Jalal, A. S. (2022). Traditional to transfer learning progression on scene text detection and recognition: a survey. *Artificial Intelligence Review, 55*(4), 3457-3502. [CrossRef]

[20] Jocher, G., Chaurasia, A., & Qiu, J. (2023). *Ultralytics YOLO* (Version 8.0.0) [Software]. Zenodo. [CrossRef]

[21] Tian, Z., Huang, W., He, T., He, P., & Qiao, Y. (2016, September). Detecting text in natural image with connectionist text proposal network. In *European conference on computer vision* (pp. 56-72). Cham: Springer International Publishing. [CrossRef]

[22] Shi, B., Bai, X., & Yao, C. (2016). 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. [CrossRef]

Yuejie Li received the Ph.D. degree in pattern recognition and intelligent system from North China Electric Power University, Beijing 102206, China, in 2023. Her research interests include the finite element methods for the various PDEs as well as their reduced-dimension methods based on the proper orthogonal decomposition, and computer applications. She served as an Associate Editor for the Journal of Numerical Simulations in Physics and Mathematics in the IECE. (Email: lyj@oit.edu.cn)

Shijun Li received his Ph.D. degree in Electrical Engineering from Hunan University, Changsha 410082, China, in 2019. He is currently an associate professor and master's supervisor, whose main research focuses on machine vision, electrical control and related fields. (Email: junshili@hnie.edu.cn)

Zhendong Luo received the Ph.D. degree in computational mathematics from University of Science and Technology of China (USTC), Hefei 230026, China, in 1997. His research interests include almost all numerical calculation methods such as the finite element method, the finite difference scheme, the finite volume method, the space-time finite method, the natural boundary element method, the spectral method, and the spectral element method, as well as their reduced-dimension methods based on the proper orthogonal decomposition, and their applications. He served as an Editor-in-Chief for the Journal of Numerical Simulations in Physics and Mathematics in the ICCK. (Email: zhdluo@ncepu.edu.cn)