Hindawi
Complexity
Volume 2019, Article ID 9345861, 11 pages
https://doi.org/10.1155/2019/9345861

Research Article

Focal CTC Loss for Chinese Optical Character Recognition on Unbalanced Datasets

Xinjie Feng, Hongxun Yao, and Shengping Zhang

Harbin Institute of Technology, China

Correspondence should be addressed to Hongxun Yao; h.yao@hit.edu.cn

Received 15 September 2018; Revised 3 December 2018; Accepted 19 December 2018; Published 2 January 2019

Guest Editor: Li Zhang

In this paper, we propose a novel deep model for unbalanced distribution Character Recognition by employing focal loss based connectionist temporal classification (CTC) function. Previous works utilize Traditional CTC to compute prediction losses. However, some datasets may consist of extremely unbalanced samples, such as Chinese. In other words, both training and testing sets contain large amounts of low-frequent samples. The low-frequent samples have very limited influence on the model during training. To solve this issue, we modify the traditional CTC by fusing focal loss with it and thus make the model attend to the low-frequent samples at training stage. In order to demonstrate the advantage of the proposed method, we conduct experiments on two types of datasets: synthetic and real image sequence datasets. The results on both datasets demonstrate that the proposed focal CTC loss function achieves desired performance on unbalanced datasets. Specifically, our method outperforms traditional CTC by 3 to 9 percentages in accuracy on average.

1. Introduction

Recently, Deep Convolutional Neural Networks (DCNN) have achieved great success in various computer vision tasks, such as image classification and object detection [1–6]. Such success is supposed to contribute to large-scale data, dropout [7], and regularization [8–12] techniques. Image sequence recognition, which can be regarded as a variant of object detection, still remains challenging due to the difficulty in detection sequence-like objects. Different from classification and detection problems which predicts one label for an entire image or a region, sequence recognition is required to compute a sequence of labels for an input image, as shown in Figure 1.

In such cases, we cannot readily apply Deep Convolutional Neural (DCNN) Networks [13, 14] to sequence-like recognition tasks since DCNN can only generate label sequences in fixed lengthes depending on the input sequences. This limitation constrains its application in scenes that require to predict sequences of various length.

Traditional methods including [15, 16] are based on a detection-recognition strategy. Individual characters are firstly detected and then recognized to form a full sentence. However, detecting a single character is challenging especially for Chinese words. Different from English, a lot of Chinese words are composed of structural parts in left-right order. This phenomenon restricts the application of detection-recognition methods.

A commonly used method is by overcutting the input sequence into slices and recognizing them through recurrent neural networks (RNN). Compared with the aforementioned detection-recognition methods, this method cuts them into many slices before feeding them into a RNN based recognition model. Due to the great power of remembering past information, it is not required to locate characters for RNN models. The final results are predicted by fusing memories into current state information. There is another challenge for Chinese image-based sequence recognition, i.e., the unbalance of training set. Different from small lexicon language datasets, large lexicon language datasets, such as Chinese, suffer from severe unbalanced sample distribution. Most words except for the small part are rarely used in everyday scenes. In this paper, we refer to the commonly used samples as *easy samples* and others as *hard samples*.

Existing methods for sequence recognition can be classified into two branches: seq2seq fashion [17, 18] and CTC

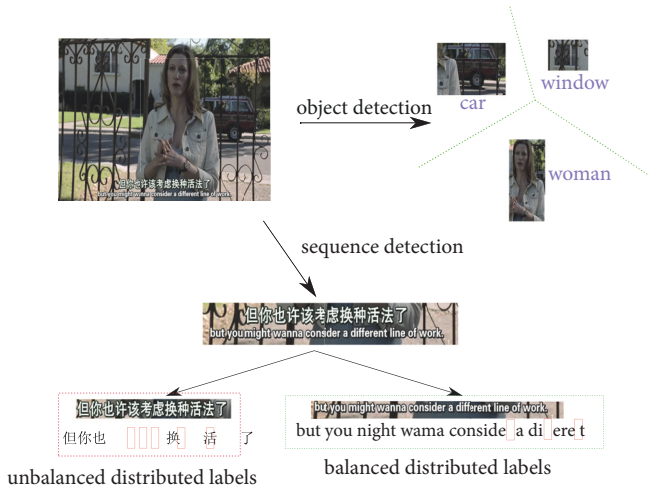

Figure 1: Distinct from object detection which aims to locate and recognize important objects, sequence detection is required to output a sequence of labels in various length. However, the prediction could be challenging if the distribution of labels is unbalanced.

loss function based models [19]. None of the above works take unbanlanced datasets into consideration, especially in Chinese image-based sequence recognition tasks. The unbanlance of a dataset will result in severe overfitting for easy samples and underfitting for hard samples. In order to remedy the unblance problem between easy and hard samples during training, we propose focal CTC loss function to prevent the model from forgetting to train the hard samples. To the best of our knowledge, this is the first work attempting to solve the unbalance problem for sequence recognition.

2. Related Work

2.1. Text Recognition Based on Manually Designed Image Features. Previous work mainly focuses on developing a powerful character classifier with manually designed image features. Wang proposed a HoG feature with random ferns developed for character classification in [20]. Neumann proposed new oriented strokes for character detection and classification in [21]. Manually designed image features always are confined to low-level which limits the performance. Lee developed a mid-level representative of characters with a discriminative feature pooling in [22]. Yao developed a mid-level feature named Strokelets to describe the parts of characters in [23]. Other interesting works brought insightful ideas by proposing to embed word images in a common vectorial subspace and convert word recognition into a retrieval problem [24, 25]. Some works take advantages of RNN which extract a set of geometrical or iamge features from handwritten texts into a sequence of image features [26]. Some other approaches [27] treat scene text recognition as an image classification problem and assign a class label to each English word (90K words in total).

2.2. CTC Loss Function Based Sequence Recognition. Connectionist Temporal Classification (CTC) is proposed in [19], which presents a CTC loss function to train RNNs to label unsegmented sequences directly. CTC is widely used for speech recognition [28, 29]. In this paper, we focus on applying CTC in image-based sequence recognition applications. Graves proposed the first attempt to combine recurrent neural networks and CTC for off-line handwriting recognition in [30]. After the revival of neural networks, deep convolutional neural networks have been devoted to image-based sequence recognition. Hasan applies Bidirectional Long-Short Term Memory (BLSTM) architecture with CTC to recognize printed Urdu texts [31]. Shi proposed a novel neural network architecture, which integrates feature extraction, sequence modeling, and transcription into a unified framework [32, 33]. Deep TextSpotter [34] trains both text detection and recognition in a single end-to-end pass. He developed a Deep-Text Recurrent Network (DTRN) that regards scene text reading as a sequence labeling problem [35].

2.3. Seq2seq Based Sequence Recognition. seq2seq and attention frameworks are prevalent in machine translation research [36]. Recently, such frameworks are adopted for image-based sequence recognition. Ba proposed a deep recurrent neural network trained with reinforcement learning to attend to the most relevant regions of an input image and the model learns to both localize and recognize multiple objects despite being given only class labels during training [37]. Lee presents recursive recurrent neural networks with attention modeling for lexicon-free optical character recognition in natural scene images [18]. Xu introduced an attention based model that automatically learns to describe the content of images [38]. Another interesting work is Spatial Transformer Networks [39], which introduce a new learnable module, the Spatial Transformer. It explicitly allows the spatial manipulation of data within the network. Focal loss was proposed to address class imbalance by reshaping the standard cross entropy loss such that it downweights the loss

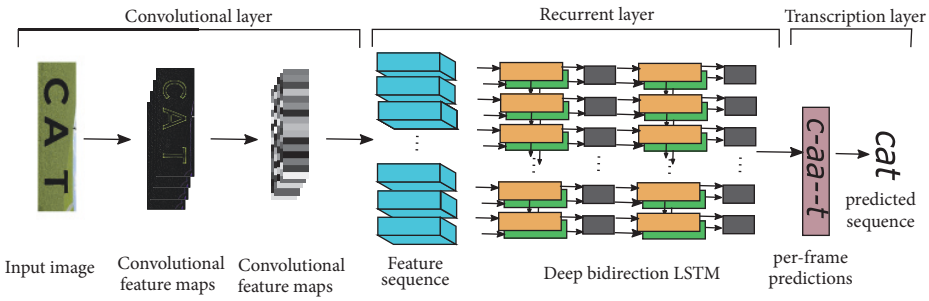

Figure 2: The network architecture. The architecture consists of three parts: convolutional layers, which extract a feature sequence from the input image; recurrent layers, which predict a label distribution for each frame; transcription layer, which translates the per-frame predictions into the final label sequence.

assigned to well-classified examples in an object detection framework [40]. Lee presents recursive recurrent neural networks with attention modeling (R2AM) for lexicon-free optical character recognition in natural scene images [41].

3. Architecture

We design our model by extending the architecture proposed by [32], which consists of three components: convolutional, recurrent, and transcription layers. The overview of the architecture is illustrated in Figure 2. We adopt the Residual Network in [42], which contains 51 layers as the convolutional layers and bidirectional Long-Short Term Memory [43–45] as the recurrent layers. The transcription layer is mainly based on CTC or our focal CTC function.

Similar to many CNN based Computer Vision applications, the convolutional layers are employed for generating feature maps of the given image. As has been already demonstrated by some detection literatures, such as Fast-RCNN and Faster-RCNN, it is possible to locate and recognize objects with feature maps from a pretrained CNN model. Hence, this strategy can be readily adopted by image-based sequence recognition task. Specifically, we first overcut the feature maps into multiple slices. Each slice can be regarded as a representation of a bounding box. These slice feature maps form a fix-length sequence. Then we feed this sequence into a RNN-based layer to predict variable-length label sequence. Note that the deep structure of ResNet is able to provide enough receptive field for a small area that cover the corresponding character. So RNN is not necessary for image-based sequence recognition mission. Despite this fact, we still utilize LSTM to predict label sequences due to its excellent ability in retaining and discarding previous information. The transcription layer is responsible for translating the output of hidden unit of LSTM to labels and finding the label sequence that has the highest probability conditioned on the per-slice predictions. We employ a fully connection layer to interface with the output of LSTM hidden units. Then we calculate the probability of each label through a softmax layer. Finally, we calculate the CTC loss through our focal loss based CTC layer with the predicted and ground truth label sequences as input.

3.1. Feature Extraction. In order to obtain a suitable representation of a given image, many CV models use pretrained CNN to generate feature maps. In fact, one of the advantages for CNNs over other traditional feature extracting methods is that they can capture local textures by different filters. For this reason, a well-trained CNN model can be readily adopted by image sequence recognition tasks. In our case, we use Residual Network, known as ResNet, to extract visual feature maps. It is first proposed by He et al. in [42] to solve classification problems and won 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation. Compared with other previously proposed deep CNN models, ResNet has deeper layers but low complexities. The elegant performance of ResNet should be contributed to the introduction of deep residual learning framework. We briefly illustrate this structure in Figure 3. The formulation of residual learning can be viewed as inserting shortcut connections into a plain feed-forward network. In fact, shortcut connections simply perform identity mapping and their outputs are added to the outputs of the following layers (Figure 3). Supposing that the input of residual learning block is denoted as X, the learning process can be formulated as optimising a residual function $F(x)$:

$$F(X) \coloneqq H(X) - X \tag{1}$$

where $H(\cdot)$ denotes an underlying mapping to be fit by several stacked layers. Note that the size of X should be consistent with the output of $H(\cdot)$.

In our experiments, the Residual Network consists of four bottleneck blocks: $9(layers) \times 16(filters)$, $12(layers) \times 32(filters)$, $18(layers) \times 64(filters)$, and $9(layers) \times 128(filters)$. Between two connected bottleneck blocks, a 1×1 filter shortcut is inserted to align the filter dimension.

We take the output of the last CNN layer of ResNet as feature maps that correspond to the entire image. We overcut this feature map into slices. Similar to Fast RCNN or Faster

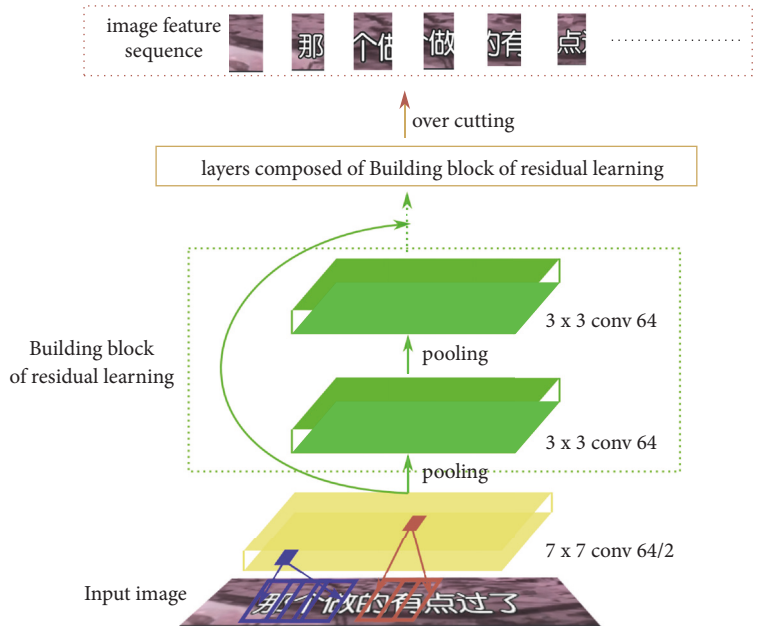

Figure 3: Convolutional layers (ResNet) which are used to extract image feature sequences. The basic building block is residual learning unit, surrounded by the green dash box.

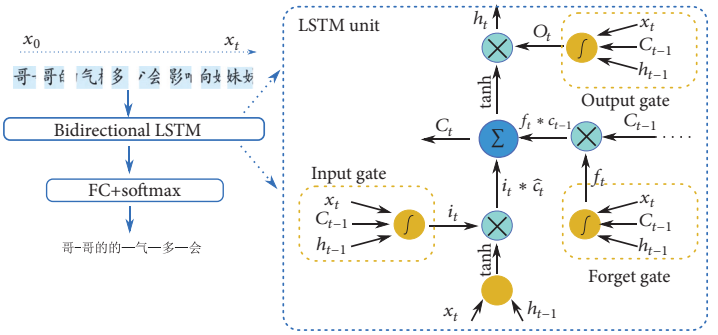

Figure 4: A schematic illustration of a LSTM neuron. Each LSTM neuron has an input gate, a forget gate, and an output gate.

RCNN, each slice contains information of a local area of the original image.

3.2. Label Sequence Prediction. Based on feature maps of image slices, we predict label sequence based on a bidirectional LSTM network. RNN [43] is a class of neural networks for efficient modeling dynamic temporal behavior of sequences through directed cyclic connections between units. Each unit is able to retain internal hidden states which are considered to contain information of previous ones. Generally speaking, RNN can be viewed as an extension of hidden Markov models. In spite of the advantages in dealing with sequential signals, traditional RNN unit suffers from the vanishing gradient problem[46], which limits the range of context it can store and thus makes training process difficult. To solve this issue, Long-Short Term Memory(LSTM), a variant of RNN, is proposed. It is capable of capturing long and short term temporal dependencies than traditional RNN unit. Specifically, LSTM extends RNN by adding three gates to the RNN neuron: a forget gate f controlling to what extent the current information is supposed to be retained; an input gate i to decide how much effect the current input should have on the hidden state; an output gate o to constrain the information of current memory which is available to output the hidden state. These gates enable LSTM to solve long-term dependency problems in sequence recognition tasks. More importantly, LSTM is easier to optimize as these gates help the input signal to effectively propagate through the recurrent hidden states $h(t)$ without affecting the output. Figure 4 is a schematic illustration of a LSTM unit. LSTM also alleviates the gradient vanishing or exploding issues of RNN [47]. In our case, we formulate the operation of a LSTM unit in (8a). For convenience, we omit the indication of forward or backward layers.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2a}$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{2b}$$

$$\widetilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{2c}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{2d}$$

$$o_t = \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{2e}$$

$$h_t = o_t * \tanh\left(C - t\right) \tag{2f}$$

where σ is the sigmoid activation function, which calculates probabilities for all gates. f_t, i_t, o_t represent the forget gate, input gate, and output gate at tth step, respectively. C_t, C_{t-1} store information of current and last cell state. h_t, h_{t-1} represent the hidden units of the two successive steps. W_* and b_* are the weights and bias, which transform two vectors into one common space. In the literature [48], rectified linear units (ReLU) are also employed as the activation function.

In our case, the label sequence is predicted through a Bidirectional LSTM. The size of hidden unit is 128. Each label is calculated by fusing hidden state of forward and backward hidden layers. At the tth step, h_t^f and h_{T-t}^b are combined through a concatenate layer followed by a full-connect layer. The final results, which take the form of probability distributions, are obtained through a softmax layer.

4. Transcription

Transcription is used to convert the predictions of each slice made by the Bidirectional LSTM into a label sequence. In this section, we first briefly review the definition of CTC loss and then introduce our proposed focal CTC loss. The CTC loss function, which is first proposed in [19], aims to model the conditional probability of label sequences given probability distribution of each predicted label. In essence, a CTC layer is supposed to be a kind of loss functions rather than a network layer. For this reason, the terminology of *CTC* layer is not accurate and may lead to misunderstandings about CTC. The focal CTC loss function is mainly inspired by the focal strategy in object detection applications. The main contribution of this paper is that by employing focal strategy, CTC loss function can be more effective in optimising the entire model.

4.1. Probability of Label Sequences. Let $\mathbb{R}$ and $\mathbb{L}$ be a real number set and a label set, respectively, which are always named as lexicon. Let $\mathcal{X} = \mathbb{R}^{m \times t}$ be the feature space of the input, and $\mathcal{Y} = \mathbb{L}^s$ be the label space, where superscripts m, t, s represent feature dimension, sequence time, and label length, respectively. Following the previous method, the input is overcut into slices. Each slice is supposed to contain a fraction of some single label characters, implying $t \geq s$. CTC loss function can be regarded as modeling the joint probability distribution over $\mathcal{X}$ and $\mathcal{Y}$, which is denoted as $\mathcal{D}_{\mathcal{X} \times \mathcal{Y}}$.

A CTC loss function has an input of a softmax layer [49]. We add a blank label $\mathbf{B}$ to $\mathbb{L}$ and hence obtain a new label $\mathbb{L}^+ = \mathbb{L} \cup \{\mathbf{B}\}$. An input sequence $\mathbf{x} \in \mathbb{R}^{m \times T}$ is transformed to another sequence $\mathbf{y} \in \mathbb{L}^{+T}$ through the softmax layer. We denote activation of output unit k at time t as y_k^t. Then y_k^t is interpreted as the probability of observing label k at time t, which defines a distribution over the set $\mathbb{L}^{+T}$ of length T sequences of the lexicon $\mathbb{L}^+ = \mathbb{L} \cup \{\mathbf{B}\}$. Reference [19] refers to the elements of $\mathbb{L}^{+T}$ as paths and denotes them as π. We assume that the distribution of the outputs of the network is conditionally independent. Then the probability of path π can be expressed as follows:

$$P\left(\pi \mid x\right) = \prod_{t=1}^{T} y_{\pi_t}^t \tag{3}$$

A sequence-to-sequence mapping function $\mathcal{B}$ is defined on sequence $\pi \in \mathbb{L}^{+T}$. $\mathcal{B}$ maps π onto l by firstly removing the repeated and blank labels. For example, $\mathcal{B}$ maps "**B**1**BB**1**B**220" onto "1120". The conditional probability is then calculated through summing probabilities of all π that are mapped by $\mathcal{B}$ onto l:

$$P\left(l \mid \mathbf{y}\right) = \sum_{\pi:\mathcal{B}(\pi)=l} p\left(\pi \mid \mathbf{y}\right) \tag{4}$$

The time complexity of the naive way to compute the conditional probability of (4) is exponential as $sizeof(\mathbb{L}^+)^T$ paths exist. Reference [19] provides an efficient dynamic programming algorithm to compute the conditional probability. The CTC loss is obviously differentiable since the conditional probability $P(l \mid \mathbf{y})$ only contains addition and multiplication operations.

4.2. Focal CTC Loss. In [40], the cross entropy of focal loss is defined as follows:

$$FL\left(p_t\right) = -\alpha_t \left(1 - p_t\right)^\gamma \log\left(p_t\right) \tag{5}$$

where p_t is the probability of ground truth in the softmax output distribution. α and γ are hyperparameters used to balance the loss. An intuitive understanding of the focal loss is that it can be viewed as multiplying cross entropy by $\alpha_t(1 - p_t)^\gamma$ (the minis belong to cross entropy: $-\log(p_t)$). It is easy to find that the closer p_t approaches to 1, the smaller the focal loss will be. So the focal loss will reduce the effect of examples but pay more attention to hard negative samples during training.

With the focal theory, we redefine our focal CTC loss as follows:

$$F\_CTC\left(l \mid \mathbf{y}\right) = -\alpha_t \left(1 - P\left(l \mid \mathbf{y}\right)\right)^\gamma \log\left(P\left(l \mid \mathbf{y}\right)\right) \tag{6}$$

where $(P(l \mid \mathbf{y})$ is the conditional probability mentioned above. The negative log function converts the optimization process from a maximization problem to a minimization problem in order to adopt gradient decent algorithm. In this way, we can focus our loss on undertrained samples and "ignore" overtrained samples.

5. Evaluation

5.1. Datasets. In this section, we evaluate the focal CTC loss on both synthetic and real datasets. We establish two synthetic datasets by concatenating 5 MNIST [13] images which are resized to 32×32 in y-axis to a long image with a resolution of 32×160. We split the alphabet "$0-9a-z$" into two subalphabets "$0-9a-h$" and "$i-z$" with the same size. The

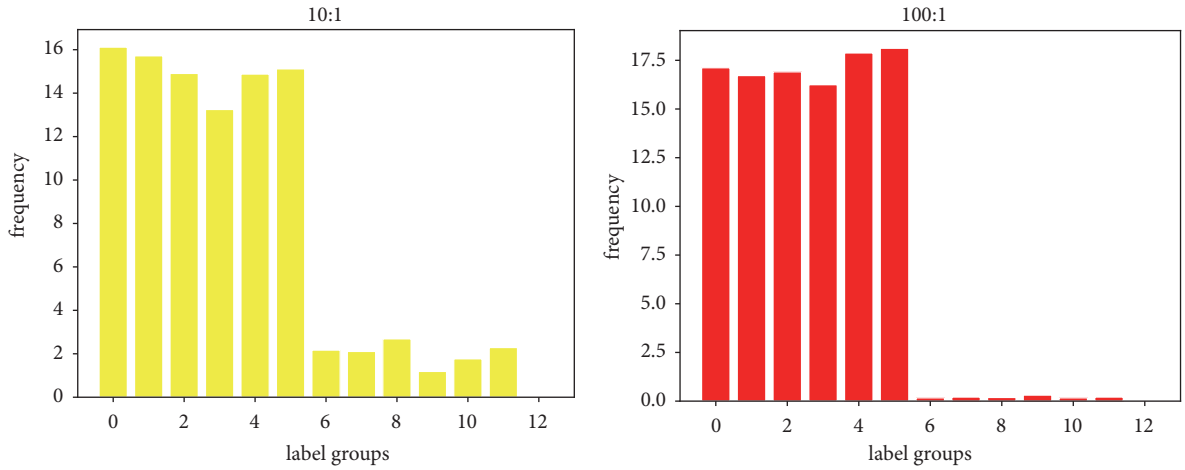

Figure 5: We provide distribution of labels for both two synthetic datasets. Each bar represents the frequency of two characters. For example, the first bar of (a) represents how many 0 and 1 exist in dataset with unbalance ratio 10 : 1.

first dataset with a unbalance ratio of 10:1 consists of two parts, one containing 1,000,000 long images which are concatenated by 5 images randomly sampling from "0 − 9a − h" and the other one containing 100,000 long images concatenated by 5 images randomly from "i − z". The second dataset with an unbalance ratio of 100:1 consists of 1,000,000 long images which are concatenated by 5 images randomly sampling from "0 − 9a − h" and 10,000 long images concatenated by 5 images randomly from "i − z". We use a dataset containing 10,000 images to test the accuracy. The ratio of high-frequency and low-frequency characters we used in training phrase is set to be 1 : 1. We present label distribution of both datasets in Figure 5.

We also test the focal CTC loss on a real Chinese-ocr dataset [50], which consists of 3,607,567 training and 5,000 test samples. Each of them is a 32 × 280 pixel image with a 10- -Chinese-character label. The frequencies of words are illustrated in Figure 6.

5.2. Training Strategy Evaluation Metrics. We implement our focal loss function in tensorflow framework, which is known as a flexible architecture supporting complex computations in machine learning and deep learning. A typical CTC loss function can be formulated as follows:

$$f_{CTC\_loss}(l, y, s) = -\log(P(l \mid \mathbf{y})) \tag{7}$$

where l and y denote label sequences from ground truth and output by RNN units, respectively. Both are constrained in length by an integer scalar s. This function has already been implemented in Tensorflow framework. So we first easily compute $P(l \mid y)$ by calling (7) defined in TF. Then we calculate focal loss according to (5). We summarize the entire process as follows:

$$ctc\_loss = f_{CTC_{loss}}(l, y, s) \tag{8a}$$

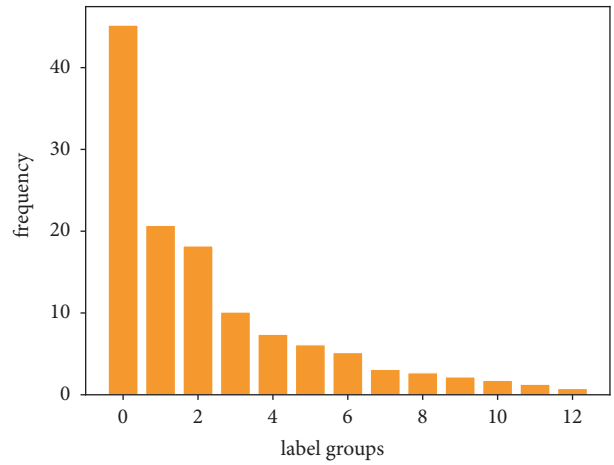

Figure 6: We provide distribution of labels for Chinese-ocr dataset. The first bar represents the most 300 frequently used words in dataset. Obviously, most Chinese words only account for a small part of all words.

$$p = e^{-ctc\_loss} \tag{8b}$$

$$focal_{loss} = \alpha \times (1 - p)^{\gamma} \times ctc\_loss \tag{8c}$$

Before we train our model, we set the learning rate and batch size to be 0.001 and 128, respectively. All parameters except CNN are initialized by sampling from a Gaussian distribution. The weights of CNN are copied from ResNet and kept unchanged during training. We optimise our model using stochastic gradient descent (SGD) with Nesterov momentum [51] set to be 0.9. We run all the experiments on a single NVIDIA M40 GPU. The entire training process is described in Algorithm 1.

We evaluate of our model in terms of two metrics: the naive accuracy and soft_accuracy. The naive one means that the predicted sequence can only be recognized as positive when it is just the same as the ground truth. The soft_accuracy

Require Parameters of RNN: Θ, learning rate $\tau = 0.001$, batch size $B = 128$

1: **for** $i = 1; i < max\_iteration; i++$ **do**
2: $V = getBatch(train\_setm, B)$;
3: $logits = Forward(V)$;
4: $ctc\_loss = tf.nn.ctc\_loss(labels, logits)$
5: $p = tf.exp(-ctc\_loss)$
6: $focal\_loss = \alpha \times (1-p)^{\gamma} \times ctc\_loss$
7: $\nabla focal\_loss(\Theta) = Backward(V, focal\_loss)$;
8: $\Theta^{i} = \Theta^{i-1} - \tau\left(\frac{1}{B}\nabla focal\_loss\,(\Theta)\right)$
9: **end for**

Algorithm 1: Focal CTC implemented in tensorflow.

Table 1: Dataset with unbalance ratio 100:1.

α	γ	accuracy	HF	LF
CTC		**0.538**	**0.739**	**0.337**
0.99	0.5	0.587	0.753	0.421
0.99	1	0.531	0.765	0.296
0.99	2	0.511	0.742	0.280
0.75	**0.5**	**0.628**	**0.755**	**0.501**
0.75	1	0.538	0.709	0.368
0.75	2	0.501	0.704	0.297
0.5	0.5	0.525	0.741	0.310
0.5	1	0.500	0.731	0.269
0.5	2	0.504	0.722	0.287
0.25	***0.5***	***0.614***	***0.731***	***0.498***
0.25	1	0.590	0.728	0.451
0.25	2	0.508	0.685	0.331

Table 2: Dataset with unbalance ratio 10:1.

α	γ	accuracy	HF	LF
CTC		**0.657**	**0.711**	**0.603**
0.99	0.5	0.667	0.721	0.613
0.99	1	0.636	0.729	0.543
0.99	2	0.703	0.745	0.662
0.75	0.5	0.684	0.709	0.659
0.75	1	0.641	0.715	0.567
0.75	2	0.631	0.716	0.546
0.5	0.5	0.667	0.741	0.593
0.5	1	0.680	0.722	0.638
0.5	2	0.682	0.728	0.635
0.25	***0.5***	***0.723***	***0.753***	***0.694***
0.25	**1**	**0.724**	**0.751**	**0.697**
0.25	2	0.707	0.763	0.650

refers to tolerating an edit distance of 1 from prediction to label. The edit distance between two sequences p and q is defined as the minimum number of insertions, substitutions, and deletions required to change p into q.

5.3. Results. Results for various α and γ are shown in Tables 1 and 2 for the synthetic dataset with unbalanced ratios 100:1 and 10:1, respectively. The best gains of the two datasets, as highlighted in bold in Tables 1 and 2, are 9% and 6.7%, respectively. Moreover, the focal CTC not only improves the Low-frequency accuracy, which is a natural result, but also enhances the High-frequency accuracy in the 10:1 dataset. The improvement for 100:1 dataset is mainly due to the enhancement of Low-frequency data. However some choices

Table 3: Real dataset.

α	γ	accuracy	soft_accuracy
CTC		**0.723**	**0.926**
0.99	0.5	0.730	0.933
0.99	**1**	**0.764**	**0.946**
0.99	2	0.631	0.913
0.75	0.5	0.692	0.918
0.75	1	0.724	0.926
0.75	2	0.704	0.930
0.5	0.5	0.733	0.936
0.5	1	0.655	0.908
0.5	2	0.641	0.913
0.25	***0.5***	***0.759***	***0.937***
0.25	1	0.690	0.926
0.25	2	0.667	0.919

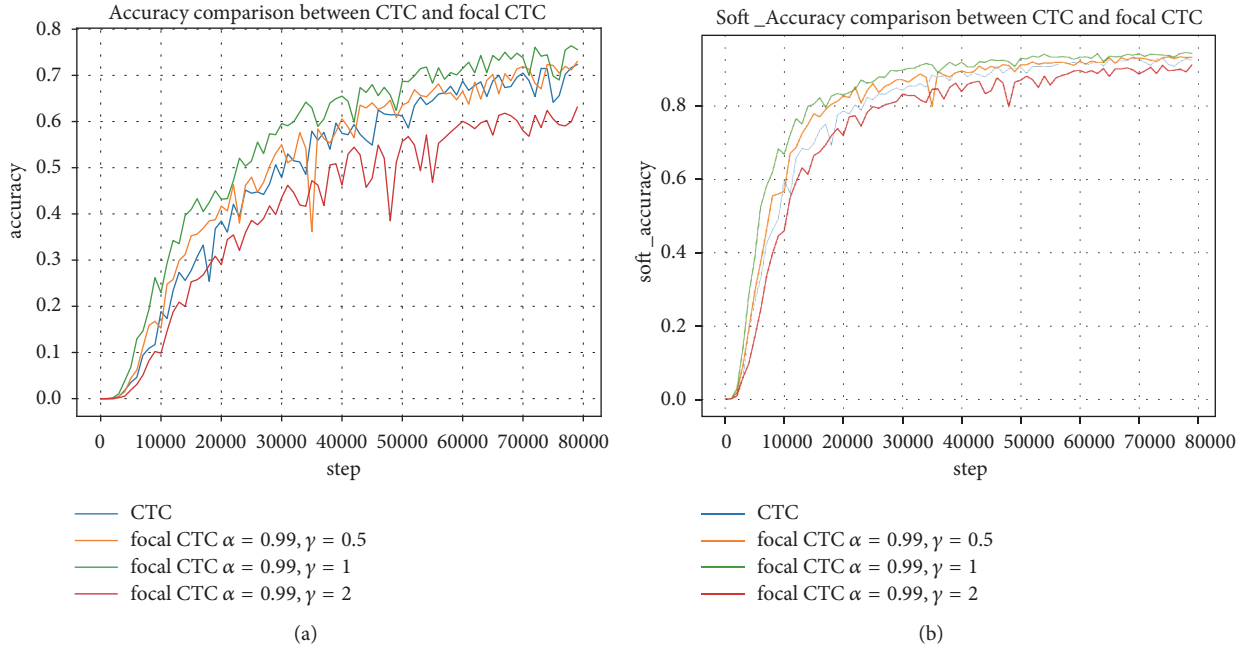

Figure 7: The comparison of convergence speed for different γ of real data with the same $\alpha = 0.99$.

of α and γ perform poor bad such as $\alpha = 0.5, \gamma = 2$ $\alpha = 0.75, \gamma = 2$ and $\alpha = 0.5, \gamma = 1$ for 100:1 dataset. For $\alpha = 0.75$, accuracy of High-frequency drops dramatically. As for $\alpha = 0.5$, accuracy of Low-frequency is not so good. The accuracy of 10:1 dataset achieves promising results for both High-frequency and Low-frequency samples. Similarly, the same issue occured in 100:1 dataset. In addition, some choices of α and γ also result in poor performance such as $\alpha = 0.99, \gamma = 1$ $\alpha = 0.75, \gamma = 1$, and $\alpha = 0.75, \gamma = 2$.

However, a bad choice of α and γ would impair the accuracy as finding that $\alpha = 0.25, \gamma = 0.5$ achieves an exiting promotion of at least 5% on both datasets, despite the existence of both High-frequency and Low-frequency data. We highlighted these results in bold italic font in Tables 1 and 2.

We present the results on the real dataset in Table 3. The best improvement of accuracy, as highlighted in bold, is 4.1%. This is an exciting improvement for a real life application. Additionally, we observe that $\alpha = 0.25, \gamma = 0.5$ makes a promotion of 3.6% which is also a considerable enhancement.

In order to observe the convergence situation for different γ, we plot the test Accuracy and Soft_Accuracy by changing curve of the real dataset for $\alpha = 0.99$. We can see that, for all training process, the focal CTC loss of $\gamma = 0.5$ achieves the best convergence ratio for both the Accuracy and Soft_Accuracy, as shown in Figure 7. The focal CTC loss of $\gamma = 2$ performs bad all the time.

With the above results on both the synthetic and real datasets, we can conclude that the focal CTC loss with $\alpha = 0.25$, and $\gamma = 0.5$ gives a considerable improvement compared with the CTC loss. Some other choices of α and γ may achieve more considerable enhancement. So in real

Figure 8: We present prediction results of Chinese word in (a). We also show some examples of synthetic datasets with unbalance ratios 10 : 1 and 100 : 1 in (b) and (c), respectively.

life applications, we can choose $\alpha = 0.25$ and $\gamma = 0.5$ for unbalanced datasets.

5.4. Qualitative Results. We provide some examples of both synthetic and real datasets in Figure 8. The sequences predicted by CTC or focal CTC are marked in red and green, respectively. All images are sampled from test split. Generally, the prediction is obviously improved with focal CTC loss employed. Due to the extreme unbalance of distribution in Chinese words, it can be seen from Figure 8(a) that some uncommonly used words can not effectively be detected by CTC based model but by our proposed focal CTC based model. As for synthetic dataset, it is interesting that CTC and focal CTC both work well for 10:1 dataset. However, the performance of CTC drops in case that we make the distribution of characters more unbalanced.

6. Conclusion

In this paper, we propose a focal CTC loss function, which can balance the loss between easy and hard samples during training. We test various hyperparameters α and γ on both the synthetic and real datasets. The results show that setting $\alpha = 0.25$ and $\gamma = 0.5$ achieves a considerable improvement for both the synthetic and real dataset. Besides, we also point out that some choices may result in bad performance. To some extent, our proposed focal CTC loss function alleviates the unbalance of big lexicon sequence recognition.

Data Availability

The datasets used in the experiment are from previously reported studies and datasets, which have been cited.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partly supported by the Science and Technology Funding of China (no. 61772158 and no. 61472103) and the Science and Technology Funding Key Program of China (no. U1711265).

References

[1] L. Zhang, A. A. Mohamed, R. Chai, B. Zheng, and S. Wu, "Automated deep-learning method for whole-breast segmentation in diffusion-weighted breast mri," in *Medical Imaging*, SPIE, 2019.

[2] L. Zhang, R. Chai, S. W. Dooman Arefan, and J. Sumkin, "Deep-learning method for tumor segmentation in breast dce-mri," in *Medical Imaging*, SPIE, 2019.

[3] D. Xie, L. Zhang, and L. Bai, "Deep learning in visual computing and signal processing," *Applied Computational Intelligence and Soft Computing*, vol. 2017, Article ID 1320780, 13 pages, 2017.

[4] Z. Zhou, J. Shin, L. Zhang, S. Gurudu, M. Gotway, and J. Liang, "Fine-tuning convolutional neural networks for biomedical image analysis: Actively and incrementally," in *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 4761–4772, USA, July 2017.

[5] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, "Road crack detection using deep convolutional neural network," in *Proceedings of the 23rd IEEE International Conference on Image Processing, ICIP 2016*, pp. 3708–3712, Phoenix, AZ, USA, September 2016.

[6] Y. Qi, S. Zhang, L. Qin et al., "Hedging deep features for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[7] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[8] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 58, no. 1, pp. 267–288, 1996.

[9] P. Bühlmann and S. van de Geer, *Statistics for High-Dimensional Data*, Springer Series in Statistics, Springer, New York, NY, USA, 2011.

[10] N. M. Nasrabadi, "Pattern Recognition and Machine Learning," *Journal of Electronic Imaging*, vol. 16, no. 4, p. 049901, 2007.

[11] A. Christmann and D.-X. Zhou, "On the robustness of regularized pairwise learning methods based on kernels," *Journal of Complexity*, vol. 37, pp. 1–33, 2016.

[12] Abhishake and S. Sivananthan, "Multi-penalty regularization in learning theory," *Journal of Complexity*, vol. 36, pp. 141–165, 2016.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[15] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR '12)*, pp. 3304–3308, November 2012.

[16] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proceedings of the 2013 14th IEEE International Conference on Computer Vision, ICCV 2013*, pp. 785–792, Australia, December 2013.

[17] Y. Deng, A. Kanervisto, and A. M. Rush, "What you get is what you see: a visual markup decompiler," 2016, https://arxiv.org/abs/1609.04938v1.

[18] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for OCR in the wild," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 2231–2239, USA, July 2016.

[19] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks," in *Proceedings of the ICML 2006: 23rd International Conference on Machine Learning*, pp. 369–376, USA, June 2006.

[20] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV '11)*, pp. 1457–1464, IEEE, Barcelona, Spain, November 2011.

[21] L. Neumann and J. Matas, "Scene text localization and recognition with oriented stroke detection," in *Proceedings of the 14th IEEE International Conference on Computer Vision (ICCV '13)*, pp. 97–104, December 2013.

[22] C.-Y. Lee, A. Bhardwaj, W. Di, V. Jagadeesh, and R. Piramuthu, "Region-based discriminative feature pooling for scene text recognition," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, pp. 4050–4057, USA, June 2014.

[23] X. Bai, C. Yao, and W. Liu, "A learned multi-scale representation for scene text recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4042–4049, 2014.

[24] J. Almazan, A. Gordo, A. Fornes, and E. Valveny, "Word spotting and recognition with embedded attributes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 12, pp. 2552–2566, 2014.

[25] J. A. Rodriguez-Serrano, A. Gordo, and F. Perronnin, "Label Embedding: A Frugal Baseline for Text Recognition," *International Journal of Computer Vision*, vol. 113, no. 3, pp. 193–207, 2015.

[26] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[27] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *International Journal of Computer Vision*, vol. 116, no. 1, pp. 1–20, 2016.

[28] A. Hannun, C. Case, J. Casper et al., "Deep speech: Scaling up end-to-end speech recognition," https://arxiv.org/abs/1412.5567.

[29] D. Amodei, S. Ananthanarayanan, R. Anubhai et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, pp. 173–182, 2016.

[30] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems, NIPS 2008*, pp. 545–552, Canada, December 2008.

[31] A. Ul-Hasan, S. B. Ahmed, F. Rashid, F. Shafait, and T. M. Breuel, "Offline printed urdu nastaleeq script recognition with bidirectional LSTM networks," in *Proceedings of the 12th International Conference on Document Analysis and Recognition, ICDAR 2013*, pp. 1061–1065, USA, August 2013.

[32] B. Shi, X. Bai, and C. Yao, "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 11, pp. 2298–2304, 2017.

[33] B. Shi, X. Bai, and C. Yao, "An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition," CoRR abs/1507.05717, https://arxiv.org/abs/1507.05717.

[34] M. Busta, L. Neumann, and J. Matas, "Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework," in *Proceedings of the 16th IEEE International Conference on Computer Vision, ICCV 2017*, pp. 2223–2231, Italy, October 2017.

[35] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang, "Reading scene text in deep convolutional sequences," in *Proceedings of the 30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pp. 3501–3508, USA, February 2016.

[36] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," https://arxiv.org/abs/1409.0473.

[37] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," https://arxiv.org/abs/1412.7755.

[38] K. Xu, J. L. Ba, R. Kiros et al., "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pp. 2048–2057, France, July 2015.

[39] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proceedings of the 29th Annual Conference on Neural Information Processing Systems, NIPS 2015*, pp. 2017–2025, Canada, December 2015.

[40] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, Venice, October 2017.

[41] C.-Y. Lee and S. Osindero, "Recursive recurrent nets with attention modeling for ocr in the wild," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778, July 2016.

[43] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[44] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of Machine Learning Research*, vol. 3, no. 1, pp. 115–143, 2003.

[45] J. Zhou and W. Xu, "End-to-end learning of semantic role labeling using recurrent neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1127–1137, Beijing, China, July 2015.

[46] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 5, no. 2, pp. 157–166, 1994.

[47] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*, pp. 2347–2355, USA, June 2013.

[48] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *Proceedings of the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '13)*, pp. 8609–8613, May 2013.

[49] J. S. Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing (Les Arcs, 1989)*, vol. 68, pp. 227–236, Neurocomputing, Springer, Berlin, Germany, 1990.

[50] Y. Chenguang, "chinese_ocr," 2018, https://github.com/YCG09/chinese_ocr.

[51] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On The Importance of Initialization And Momentum in Deep Learning," in *International Conference on Machine Learning*, pp. 1139–1147, June 2013.