An Efficient Prototype-Based Model for Handwritten Text Recognition with Multi-loss Fusion

Ming-Ming Yu[1,2(✉)], Heng Zhang[1], Fei Yin[1], and Cheng-Lin Liu[1,2]

[1] National Laboratory of Pattern Recognition (NLPR), Institution of Automation, Chinese Academy of Sciences, Beijing 100190, China
{fyin,liucl}@nlpr.ia.ac.cn

[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
{yumingming2020,heng.zhang}@ia.ac.cn

Abstract. Prototype learning has achieved good performance in many fields, showing higher flexibility and generalization. In this paper, we propose an efficient text line recognition method based on prototype learning with feature-level sliding windows for classification. In this framework, we combine weakly supervised discrimination and generation loss for learning feature representations with intra-class compactness and inter-class separability. Then, dynamic weighting and pseudo-label filtering are also adopted to reduce the influence of unreliable pseudo-labels and improve training stability significantly. Furthermore, we introduce consistency regularization to obtain more reliable confidence distributions and pseudo-labels. Experimental results on digital and Chinese handwritten text datasets demonstrate the superiority of our method and justify advantages in transfer learning on small-size datasets.

Keywords: Text recognition · Prototype learning · Consistency regularization · Connectionist temporal classification · Pseudo-label

1 Introduction

Text recognition has drawn much research interest in the computer vision community due to its wide applications. The text recognition problem is to map the input image to the corresponding sequence of characters. With the development of deep learning, excellent performance is achieved in many scenarios, such as scene text recognition and handwritten text recognition. Based on the encoder-decoder framework, visual features are usually extracted by the CNN (Convolutional Neural Networks) encoder, followed by the RNN (Recurrent Neural Networks) or Transformer network to extract context features. Finally, the Connectionist Temporal Classification (CTC) [5] or attention mechanism is used to align feature sequences and labels. Some other works [9,34] remove the sequence modeling stage and only use the CNN encoder, improving the parallelization

U. Porwal et al. (Eds.): ICFHR 2022, LNCS 13639, pp. 404–418, 2022.
https://doi.org/10.1007/978-3-031-21648-0_28

and reducing the computation cost. Text recognition methods have achieved high performance, but the interpretability and generalization are still inadequate [3,31]. Different from discriminative models, the Convolution Prototype Network (CPN) [31,32] is a discriminative and generative hybrid model with better generalization and robustness. CPN has been successfully applied to open set recognition and few-shot learning in character classification tasks. However, it is challenging to apply CPN on weakly-supervised text line recognition because a text line is composed of different characters with variable-length and unknown character positions.

To address the abovementioned issues, we propose a prototype classifier for text recognition to improve the generalization and robustness based on weakly-supervised discrimination and generation learning. Our model adopts the convolution-CTC framework with sliding windows at the feature level. The blank class can be considered as all the non-character samples dynamically generated in the window sliding process, so we also set a prototype for the blank class to fully use these non-character samples and enhance the discriminant of the model. The blank prototype is only used in discriminative learning rather than generative learning. Dynamic weighting and pseudo-label filtering are also adopted to weaken unreliable pseudo-labels and improve training stability. Moreover, consistency regularization [13,25] is introduced to obtain more reliable confidence distributions and pseudo-labels for improving the performance. The experimental results show that the AR and CR of our method can reach 93.66% and 93.90%, respectively, on the ICDAR-2013 dataset without a language model. Moreover, the string accuracy is 95.53% and 95.49% on CAR-A and CAR-B test sets, respectively. We also demonstrate that our model can better transfer the knowledge learned from the isolated character data to the text line, with obvious performance gain.

Our main contributions are summarized as follows: (1) We design a prototype-based text line recognition method with feature-level sliding windows for classification. Our feature-level sliding window method shows less memory footprint and higher inference speed than sliding windows on the input image. (2) In order to obtain robust pseudo-labels, we propose two methods to weaken unreliable pseudo-labels: pseudo-label filtering and dynamic weighting. Moreover, we introduce consistency regularization to enhance model robustness and further improve recognition performance. (3) Our method has achieved state-of-the-art performance on three handwritten text datasets and justifies the advantages in transfer learning from character recognition to text recognition with small-size datasets.

2 Related Work

2.1 Text Recognition

The existing text line recognition methods can be divided into two methods based on explicit and implicit segmentation. In the explicit segmentation approach, the most representative one is recognition based on the over-

segmentation strategy [21,24,27]. Specifically, these methods first generate candidate characters by merging continuous primitives and then search for the optimal segmentation-recognition path in the candidate lattice. Finally, the score of the segmentation-recognition path is given by integration of the character classifier, language model, and geometric models. With only string-level annotations, Wang et al. [24] proposed a weakly supervised training strategy for character classifier training under the over-segmentation framework. Moreover, Peng et al. [11] formulate a new segmentation-based text recognition framework for segmenting and recognizing characters end-to-end. The models based on implicit segmentation do not need to perform explicit character segmentation and only use string-level annotations, including Hidden Markov Model (HMM) [2], CTC, Attention. With the development of deep learning, CTC and attention-based methods have been widely used in text recognition. Grave et al. [5] first applied CTC to handwritten text line recognition with RNN for contextual feature representation. Shi et al. [14] proposed the CRNN (Convolution Recurrent Neural Networks) model combining the feature extraction capability of CNN with the sequence modeling capability of RNN. Yin et al. [34] proposed a fully convolutional model with multi-scale sliding window classification. Liu et al. [9] proposed context beam search to combine the Transformer-based language model with a visual model. On the other hand, attention-based text recognition uses a 1D soft-attention model to select relevant local features during character decoding. In this manner, the model can learn a character-level language model from the training data. Shi et al. [15] proposed an end-to-end framework with RNN and attention for scene text recognition. SAR [8] used the 2D attention to recognize irregular texts. Wang et al. [19] employed the Transformer to replace RNN structure to capture long distance context.

2.2 Prototype Learning

Prototype learning is a method to classify samples based on template matching. The prototype (template) refers to a representative point in the sample or feature space. K-Nearest-Neighbor (KNN) and Learning Vector Quantization (LVQ) [6] are both classical prototype classifiers. Traditional prototype models cannot be optimized end-to-end, because feature extraction and prototype learning are performed in separate stages. With the development of deep learning, some works combined DNN with prototype learning to improve model performance. Snell et.al [16] applied the prototype concept in CNN for few-shot learning. Yang et al. [31,32] proposed the Convolutional Prototype Network (CPN) by training the CNN feature extractor and prototypes end-to-end. With the supervision of discriminative and generative learning, the CPN shows better robustness and generalization in multiple scenarios such as open set recognition and few-shot learning. Gao et al. [3] first proposed a prototype-based handwritten text recognition method; however, it still suffers from unreliable pseudo-labels and the high computation cost caused by sliding windows on the input image.

2.3 Consistency Regularization and Pseudo-labels

Consistency regularization and pseudo labels are usually used in semi-supervised learning. Based on the assumption that the model should output consistent probability distribution for the same input with slight disturbances, consistency regularization [13,18,29] can minimize the difference between prediction distributions from different disturbances of the same data. Random data augmentation, dropout [17], and exponential moving average (EMA) [13,18] are commonly used to add disturbances. Kullback-Leibler Divergence, Jensen-Shannon Divergence, cross-entropy, and mean square error (MSE) are frequently-used methods to measure the difference between two probability distributions. After training an initial model on a few labeled data, pseudo-labels [7] are given by the prediction on unlabeled data. For handwritten recognition, Gao et al. [4] use pseudo-labels in the CTC loss and get state-of-art results.

3 Method

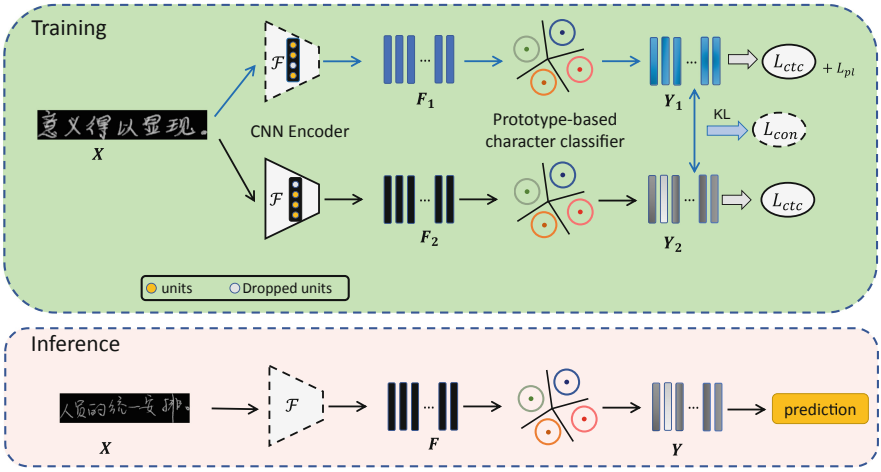

Fig. 1. An illustration of the proposed text recognition method based on prototype learning.

The overall framework of the proposed method is illustrated in Fig. 1. As the figure shows, we use the CTC-based convolutional prototype learning framework. The text image is first normalized by scaling to a fixed height with the aspect ratio preserved. Then the normalized text image is encoded by CNN for feature representation of candidate characters. After that, the candidate character features are classified by the prototype-based character classifier. In prototype learning (PL), the input text image will go through the training model twice

with different dropout for consistency regularization computation. The two different sub-models are shown in Fig. 1. Besides, the CTC loss L_{ctc} and PL loss L_{pl} are also computed and combined with consistency regularization loss L_{con}. In prediction, we use the CTC decoding algorithm to obtain the final recognition results.

3.1 CNN Encoder and Prototype Classifier

CNN Encoder. Given a normalized text line image $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$, the encoder $\mathcal{F}$ first produces the feature map $\mathbf{F} \in \mathbb{R}^{H' \times W' \times D'}$, where H', W', and D' denote the height, width, and channel number of the feature map, respectively. Then the feature sequence $\mathbf{F}$ with L elements is formed by sliding windows on the feature map. In our experiments, the window width and height are both set to H', and the feature dimensionality is $H' \times H' \times D'$. As illustrated in Fig. 2, each feature vector in the feature sequence corresponds to a local region in the original image through the receptive field. The whole process of feature extraction can be formalized as:

$$\mathbf{F} = \mathcal{F}(\mathbf{X}) = \{\mathbf{f}^1, \mathbf{f}^2, ...\mathbf{f}^L\}. \tag{1}$$

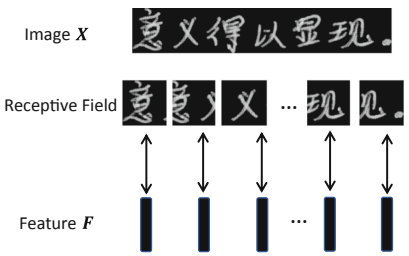

Fig. 2. Sliding windows on the feature level. Each vector is associated with a receptive field on the input image.

Prototype Classifier. We set $K + 1$ prototypes for characters and the blank in this paper, where K is the number of character categories. The prototypes are denoted as $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{K+1}\}$, $\mathbf{c}_i \in \mathbb{R}^d$. The blank in CTC can be denoted as the non-characters in the sliding window-based recognition framework. In order to make full use of non-character samples in the text lines and enhance the discriminant of the model, we set a prototype for the blank class. Following [22, 32], under the assumption of Gaussian distribution with equal identity covariance matrix, the negative euclidean distance can be used to measure the similarity between the feature $\mathbf{f}^t$ and the prototype $\mathbf{c}_k$. And the posterior probability that the t-th feature $\mathbf{f}^t$ belongs to category k can be defined as:

$$y_k^t = \frac{e^{-\gamma \|\mathbf{f}^t - \mathbf{c}_k\|_2^2}}{\sum_{k=1}^{K+1} e^{-\gamma \|\mathbf{f}^t - \mathbf{c}_k\|_2^2}}, \tag{2}$$

where γ is a hyper-parameter that controls the hardness of probability assignment. Therefore, the confidence distribution of the model output can be expressed as $\mathbf{Y} = \{\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^L\}$, where $\mathbf{y}^t \in \mathbb{R}^{K+1}$ represents the probability distribution of the t-th feature over all classes.

3.2 Weakly Supervised Discrimination Loss

Unlike character classification, text line recognition is a weakly supervised learning task, i.e., the ground truth of each feature is unknown. Therefore, we use CTC loss as our discrimination loss, which can be directly learned from sequence labels, avoiding labeling the position of each character. In the CTC recognition framework, the input is a sequence $\mathbf{Y} = \{\mathbf{y}^1, \mathbf{y}^2, ...\mathbf{y}^L\}$. The corresponding label sequence is denoted as $\mathbf{l} = \{l_1, l_2, ..., l_T\}$, where $l_i (i \in \{1, ..., T\})$ denotes the ith character and T is the total character number. The conditional probability of a path π is defined as:

$$p(\pi|\mathbf{Y}) = \prod_{t=1}^{L} y_{\pi_t}^t, \tag{3}$$

where $y_{\pi_t}^t$ is the probability of generating the character π_t at timestep t. After that, the sequence-to-sequence mapping function $\mathcal{M}$ is performed by removing the repeated characters and blanks from the given path π. The conditional probability $p(\mathbf{l}|\mathbf{Y})$ is defined as the sum of probabilities of all π which are mapped by $\mathcal{M}$ onto $\mathbf{l}$.

$$p(\mathbf{l}|\mathbf{Y}) = \sum_{\pi \in \mathcal{M}^{-1}(\mathbf{l})} p(\pi|\mathbf{Y}). \tag{4}$$

Finally, the CTC loss function L_{ctc} is defined as the negative log-likelihood of the ground-truth conditional probability:

$$L_{ctc} = -\ln p(\mathbf{l}|\mathbf{Y}). \tag{5}$$

3.3 Weakly Supervised Generation Loss

Generation loss in our work is a supervised loss and can regarded as the maximum likelihood (ML) regularization under the standard Gaussian density assumption for class-specific features. Since there are no annotations for candidate characters in each text line, we can only calculate the generation loss through pseudo-labels. This subsection describes the generation of reliable pseudo-labels and robust L_{pl} computation based on dynamic weighting and pseudo-label filtering.

Pseudo-label Generation. We adopt the soft pseudo-label distribution z_k^t of feature $\mathbf{f}^t$ as in [3,4]:

$$z_k^t = \frac{\sum_{\{\pi|\pi \in \mathcal{M}^{-1}(\mathbf{l}), \pi_t = k\}} p(\pi|\mathbf{Y})}{p(\mathbf{l}|\mathbf{Y})}, k = 0, ..., K, \tag{6}$$

where the numerator is the probabilities of feasible alignment paths through character k at time t, and the denominator is the sum of the probabilities of all feasible alignment paths. Then the soft pseudo-label distribution matrix for a text line can be represented by $\mathbf{Z}$. Compared with semi-supervised learning using predicted probabilities as pseudo-labels, our method uses the alignment rule of CTC to integrate text line label $\mathbf{l}$ and so more reliable.

Pseudo-label Filtering. We use the best path decoding to decode the pseudo-label $\mathbf{Z}$ and use the decoding results to measure the reliability of pseudo-labels. The decoding result $\mathbf{s}$ is calculated by $\mathbf{s} = \mathcal{M}(\arg\max_{\pi} p(\pi|\mathbf{Z}))$, i.e., taking π_t with the maximum probability at each time step and mapping the π onto $\mathbf{s}$ by $\mathcal{M}$. If the decoding result $\mathbf{s}$ is the same as the real label $\mathbf{l}$, the pseudo-label is considered to be reliable. Then, only the filtered reliable pseudo-labels are used to compute generation loss L_{pl}:

$$L_{pl} = \begin{cases} 0 & \text{if } \mathbf{s} \neq \mathbf{l} \\ \sum_t \sum_{k \neq blank} z_k^t \|\mathbf{f}^t - \mathbf{c}_k\|_2^2 & \text{others} \end{cases}. \tag{7}$$

Dynamic Weighting. Due to the inaccurate confidence of the model outputs in the early stage, a large number of unreliable pseudo-labels are generated. However, as the number of training epochs increases, generated pseudo-labels are relatively reliable. So we dynamically set the L_{pl} weight $\lambda(m)$, increasing with the number of training epochs:

$$\lambda(m) = \begin{cases} 0 & m \leq m_s \\ \alpha \times e^{-5(1-\frac{m-m_s}{m_{max}})} & m > m_s \end{cases}, \tag{8}$$

where m is the current epoch, α and m_{max} are two hyperparameters that control the maximum value and increment of $\lambda(m)$. When $m < m_s$, the weight is set to 0. The network ensures the accuracy of classification by discrimination loss and improves the reliability of the network confidence. When $m > m_s$, the generation loss weighting function $\lambda(m)$ ramps up, starting from zero, along a Gaussian curve. The generation loss can be regarded as a further regularization to improve the model performance.

3.4 Consistency Regularization

Consistency regularization is introduced to improve the robustness and generate more reliable pseudo-labels. Consistent regularization assumes that the consistency probability distribution should be output for the same input, although slightly disturbed. Following [25], we regard dropout as the perturbation in model learning. Specifically, the input image $\mathbf{X}$ is fed to go through the forward pass of the model twice. Thus we can get two confidence distributions, $\mathbf{Y}_1$ and $\mathbf{Y}_2$. As shown in the top part of Fig. 1, at the training phase, the dropped

units of the first path for the confidence distribution $\mathbf{Y}_1$ are different from that of the second path for distribution $\mathbf{Y}_2$, so the confidence distributions $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are different for the same input. Therefore consistency regularization constrains the consistency of the predicted probabilities by minimizing the KL divergence between two output confidence distributions for the same sample, which is defined as:

$$L_{con} = 0.5(\mathcal{D}_{KL}(\mathbf{Y}_1||\mathbf{Y}_2) + \mathcal{D}_{KL}(\mathbf{Y}_2||\mathbf{Y}_1)). \tag{9}$$

Furthermore, we also use the dynamic weighting strategy to fuse consistency regularization loss L_{con}.

Then the discrimination loss is correspondingly defined as the combination of two forward propagations:

$$L_{ctc} = -0.5(\ln p(\mathbf{l}|\mathbf{Y_1}) + \ln p(\mathbf{l}|\mathbf{Y_2})). \tag{10}$$

3.5 Total Loss

We sum the loss functions defined above. The overall objective function for training our proposed model includes three parts: weakly supervised discrimination loss L_{ctc}, weakly supervised generation loss L_{pl}, and consistency regularization loss L_{con}. The final hybrid loss is defined as :

$$L = L_{ctc} + \lambda(m)L_{pl} + \lambda(m)L_{con}, \tag{11}$$

where $\lambda(m)$ is the dynamic weight of the generation loss and consistency regularization loss for multi-loss fusion.

4 Experiments

4.1 Datasets

We conduct both Chinese and digital handwritten text recognition experiments. For handwritten Chinese recognition, we evaluate the proposed approach on ICDAR-2013 [33]. We compare our method with the state-of-the-art methods and conduct a series of ablation studies to explore the effect of each part of our models. In addition, we validate the generalization of our approach by transferring character recognition to text line recognition with small sample size. For handwritten digital text recognition, we evaluate the performance of our method on the ORAND-CAR [1].

CASIA-HWDB [10] is a large offline Chinese handwriting database, which is divided into six sub-dataset. CASIA-HWDB1.0-1.2 contain 3,118,447 isolated character samples of 7,356 classes. HWDB2.0-2.2 have 52,230 text lines, which are segmented from 5,091 handwritten pages. **ICDAR2013** competition dataset [33] includes 3,432 text lines, which are segmented from 300 handwritten pages. For Chinese datasets, we use the character samples from CASIA-HWDB 1.0-1.2 to synthesize 700,000 synthetic text images following the method proposed by Wu et al. [26]. The synthetic and real text images from HWDB2.0-2.2 are used to

train our model. **ORAND-CAR** [1] is a digital handwritten text line database. It contains 11,719 images in total, divided into CAR-A and CAR-B sub-datasets. The CAR-A database consists of 2,009 images for training and 3,784 images for testing. The CAR-B database contains 3,000 training images and 2,926 testing images.

4.2 Implementation Details

We implement experiments based on the framework of Pytorch with 4 NVIDIA RTX 24G GPUs. The architecture of CNN encoder is derived from the SeResNet1111 [9] with the residual and squeeze-and-excitation structures. We only set one prototype for each class, including the blank. All prototypes are initialized as zero vectors.

As for handwritten Chinese text recognition, the images are normalized to the height of 128 pixels and maintain their aspect ratios. The height and width of the feature map are $\frac{H}{128}$ and $\frac{W}{32}$, respectively. The Adam optimizer is applied to train our model with a learning rate initialized to 1×10^{-3}, and the learning rate will be decayed by timing 0.1 after 30 epochs. The training stops at 90 epochs. The weight decay is set to 1×10^{-4}. For the dynamic weighting in Sect. 3.3, we set m_s, m_{end} and α to 0, 60, and 0.001, respectively.

For the digital text recognition task, images are resized and padded to 32×256. Furthermore, the last three pooling layers in SeResNet1111 adopt 1×2 sized pooling windows instead of 2×2 to reduce feature dimension along the height axis only. Therefore, the shape of the feature map is $\frac{H}{32} \times \frac{W}{4} \times 512$. The model is also trained from scratch using an Adam optimizer with the base learning 1×10^{-3}, and the learning rate will be decayed by timing 0.1 after 3,000 iterations. The whole training process contains 10,000 iterations. We set m_s, m_{end} and α to 3,000, 6,000, and 0.001, respectively.

4.3 Ablation Experiments

Table 1. Recognition results of models with different training strategies on the ICDAR 2013 competition set without synthesized data and language model. (CR: correct rate; AR: accurate rate [26])

Methods	Without LM (%)	
	AR	CR
Linear Classifier + L_{ctc}	90.41	90.81
Proto + L_{ctc}	90.54	90.87
Proto + L_{ctc} + L_{pl}	90.63	90.92
Proto + L_{ctc} + L_{pl} + L_{con}	**90.96**	**91.26**

In this part, we design several variants of our model to validate the contributions of different components. We take real samples from CASIA-HWDB2.0-2.2

to train and evaluate our model on the ICDAR2013 data. In our model, K is set to 7357, including 7356 character classes in HWDB1.0-1.2 and one "unknown" token for characters not in the character dictionary. In the following subsection, we keep the same category setting.

The experimental results are shown in Table 1. Proto represents the prototype classifier. We can see that our model can achieve comparable performance with the traditional linear classifier. Furthermore, our model can perform better by introducing generation loss with dynamic weighting and pseudo-labels filtering. The weakly supervised generation loss can be used as the regularization to improve the generalization of the model. Meanwhile, dynamic weighting and pseudo-labels filtering make the weak-supervision training more stable. Consistency regularization aims to generate similar confidence distributions when the input is disturbed, thus improving the robustness of the model. Thanks to more reliable confidence distribution and pseudo-labels, performance can be further enhanced when we adopt both generation loss and consistency regularization.

4.4 Comparison with State-of-the-art Methods

Table 2. Comparison with existing methods on the ICDAR 2013 competition set. The results marked by "$*$" denotes using the powerful Transformer-based language model rather than the traditional n-gram language model, and the results marked by "$\dagger$" denotes using contextual regularization to integrate contextual information. "$\dagger*$" means using both the above two strategies.

Methods	Without LM (%)		With LM (%)	
	AR	CR	AR	CR
Wu et al. [26]	86.64	87.43	90.38	–
Wang et al. [23]	88.79	90.67	94.02	95.53
Wang et al. [24]	87.00	89.12	95.11	95.73
Gao et al. [3]	90.30	90.92	96.23	96.64
Peng et al. [12]	89.61	90.52	94.88	95.51
Xie et al. [30]	91.25	91.68	96.22	96.70
Xie et al. [28]	91.55	92.13	96.72	96.99
Liu et al. [9]	93.62	–	97.51*	–
Peng et al. [11]	93.05	93.30	–	–
Peng et al. [11]	**94.50**†	**94.76**†	**97.70**†*	**97.91**†*
Ours	93.66	93.90	97.04	97.23

In order to further improve the performance, we use synthetic and real text images for model training and experimental comparison. The comparison results with the existing methods are shown in Table 2, where we use a 5-gram statistical language model for context fusion. Without the language model, our method achieves comparable performance with AR 93.66% and CR 93.90%. The method

proposed by Peng et al. [11] performs slightly better, owing to the contextual regularization with BLSTM layers. With the language model, our approach achieves comparable performance to Liu et al. [9] and Peng et al. [11]. However, we only use the 5-gram statistical language model instead of the transformer-based language model. In addition, we also compared the parameter sizes of different methods on the ICDAR-2013 dataset, and the results are shown in Table 3. It is worth noting that we only use SeResNet1111 as the feature extractor, making a good trade-off between performance and parameter size. Especially compared with the method in [9], our method achieves similar performance with about half the size of parameters.

Table 3. The parameter size comparison with different methods.

Methods	Params (MB)	AR
Wu et al. [26]	71 MB	86.64
Liu et al. [9]	203 MB	93.62
Peng et al. [11]	119 MB	94.50
Ours	115MB	93.66

For handwritten digital recognition, the comparison of our approach with state-of-the-art methods on ORAND-CAR is shown in Table 4. Our model achieves higher accuracy and reduces the error rate by 12% compared with the previous best result, demonstrating the effectiveness of our model.

Table 4. String accuracies of different models on the digital handwritten datasets.

Methods	CAR-A	CAR-B	Average
BeiJing [1]	80.73	70.13	75.43
FSPP [20]	82.61	83.32	82.97
CRNN [14]	88.01	89.79	88.90
ResNet-RNN [35]	89.75	91.14	90.45
Gao et al. [3]	94.83	94.70	94.77
Gao et al. [4]	95.01	94.74	94.88
Our	**95.53**	**95.49**	**95.51**

Figure 3 shows some recognition results of complex samples without the language model. Our method is robust to different writing styles and slanted texts. Meanwhile, good recognition results can be achieved for punctuation marks. Failure cases are mainly due to similar glyph characters. We visualize text location in Fig. 4. In the sliding-window based text recognition framework, the character classifier can recognize characters with high scores when these characters are

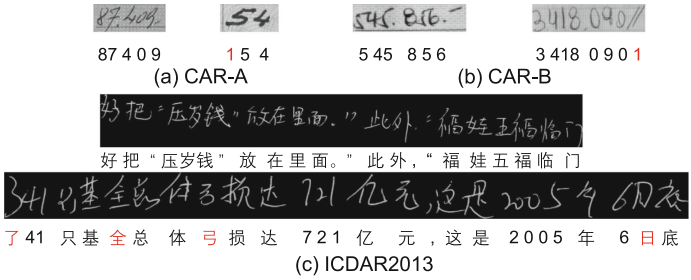

Fig. 3. Visualization of recognition results for our proposed method.

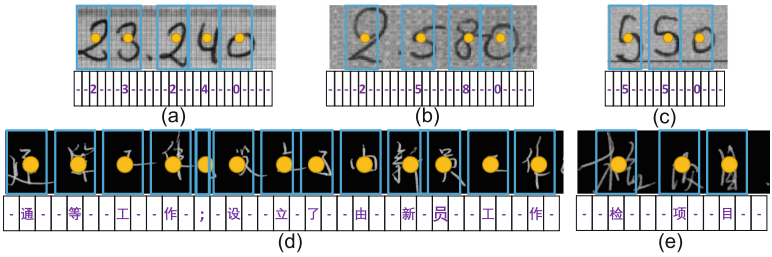

Fig. 4. Visualization of character locations, where yellow circles represent character centers. (Color figure online)

located in the center of the sliding windows. On the contrary, when the centers of sliding windows and characters are misaligned, the text recognizer will output blank labels or low character scores. Inspired by Non-Maximum Suppression, we can get the best candidate character center position by selecting the one with the highest confidence among adjacent character frames. The width of the bounding box can be obtained according to a prior of the character width. For Chinese characters in ICDAR-2013, we assume that the Chinese character width is $\frac{5}{8}$ of the image height, and for punctuation, it is $\frac{1}{4}$ of the height. For digital handwritten recognition, we assume that the digital character width is $\frac{1}{2}$ of the image height. So we can get the position of the characters by the center and width of the character bounding box.

4.5 Generalization Experiment

To demonstrate the generalization of our method, we construct a transfer learning experiment. We regard character images as short text lines to unify the character recognition and text line recognition into one framework. By sliding windows on character images, multi-frame features are generated. We can assume that the middle frame is the most aligned frame of each character and the edge frames are blank for prototype classifier training. CTC loss is uniformly used for linear classifiers trained on character and text line data. After ten epochs on character data pre-training, the linear classifier and our model can achieve comparable performance, i.e., AR being 96.15% and 96.04%, respectively. Then,

1%, 2%, 5%, and 10% real text lines are used for finetuning. Moreover, we also compare the linear classifier trained from scratch. In order to make a fair comparison, the same learning rate and batch size are used for all three models. The results are shown in Table 5. With only 1% real text lines for finetuning, our model can get 75.42% AR, 7.12% higher than finetuned linear classifier. However, when more text lines are used to finetune the model, the performance gap between the three models becomes smaller. This shows that with the increase in the number of text lines, the advantages of pre-training on isolated characters become insignificant. The experimental results show that our model has better generalization performance in transfer learning with fewer text lines. It can better apply the knowledge learned from character data to the text lines.

Table 5. AR under different percentages of training samples on the Chinese handwritten text dataset ICDAR-2013.

Sample rates (%)	*LC* (from scratch)	*LC* (finetuning)	Our (finetuning)
1	0	68.30	**75.42**
2	12.77	73.64	**78.36**
5	57.47	77.65	**80.30**
10	73.22	78.75	**80.97**

5 Conclusions

In this paper, we propose an efficient handwritten text recognition method based on the prototype classifier. By sliding windows at the feature level, our method is more efficient and can get character positions. Moreover, to improve the stability of the training and the performance of the recognition model, we propose dynamic weighting and pseudo-label filtering to weaken unreliable pseudo-labels. Furthermore, consistency regularization is used to give more reliable confidence distributions. Experimental results can demonstrate the effectiveness of our method. Furthermore, the transfer experiment from characters to text lines with small data size for fine-tuning proves that our proposed method has higher generalization.

Acknowledgements. This work has been supported by the National Key Research and Development Program Grant 2020AAA0109702, the National Natural Science Foundation of China (NSFC) grant 61936003.

References

1. Diem, M., et al.: ICFHR 2014 competition on handwritten digit string recognition in challenging datasets (HDSRC 2014). In: Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition, pp. 779–784. IEEE (2014)
2. Du, J., Wang, Z.R., Zhai, J.F., Hu, J.S.: Deep neural network based hidden Markov model for offline handwritten Chinese text recognition. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 3428–3433. IEEE (2016)

3. Gao, L., Zhang, H., Liu, C.-L.: Handwritten text recognition with convolutional prototype network and most aligned frame based CTC training. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 205–220. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_14
4. Gao, L., Zhang, H., Liu, C.L.: Regularizing CTC in expectation-maximization framework with application to handwritten text recognition. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–7. IEEE (2021)
5. Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell. **31**(5), 855–868 (2008)
6. Kohonen, T.: The self-organizing map. Proc. IEEE **78**(9), 1464–1480 (1990)
7. Lee, D.H., et al.: Pseudo-label: the simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on Challenges in Representation Learning, ICML, vol. 3, p. 896 (2013)
8. Li, H., Wang, P., Shen, C., Zhang, G.: Show, attend and read: a simple and strong baseline for irregular text recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8610–8617 (2019)
9. Liu, B., Sun, W., Kang, W., Xu, X.: Searching from the prediction of visual and language model for handwritten Chinese text recognition. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12823, pp. 274–288. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86334-0_18
10. Liu, C.L., Yin, F., Wang, D.H., Wang, Q.F.: Casia online and offline Chinese handwriting databases. In: Proceedings of the International Conference on Document Analysis and Recognition, pp. 37–41. IEEE (2011)
11. Peng, D., et al.: Recognition of handwritten Chinese text by segmentation: a segment-annotation-free approach. IEEE Trans. Multimedia (2022)
12. Peng, D., Jin, L., Wu, Y., Wang, Z., Cai, M.: A fast and accurate fully convolutional network for end-to-end handwritten Chinese text segmentation and recognition. In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 25–30. IEEE (2019)
13. Samuli, L., Timo, A.: Temporal ensembling for semi-supervised learning. In: International Conference on Learning Representations (ICLR), vol. 4, p. 6 (2017)
14. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
15. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4168–4176 (2016)
16. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. Adv. Neural Inf. Process. Syst. **30** (2017)
17. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
18. Tarvainen, A., Valpola, H.: Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results. Adv. Neural Inf. Process. Syst. **30** (2017)
19. Wang, P., Yang, L., Li, H., Deng, Y., Shen, C., Zhang, Y.: A simple and robust convolutional-attention network for irregular text recognition. arXiv preprint arXiv:1904.01375 6(2), 1 (2019)

20. Wang, Q., Lu, Y.: A sequence labeling convolutional network and its application to handwritten string recognition. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 2950–2956 (2017)
21. Wang, Q.F., Yin, F., Liu, C.L.: Handwritten Chinese text recognition by integrating multiple contexts. IEEE Trans. Pattern Anal. Mach. Intell. **34**(8), 1469–1481 (2011)
22. Wang, Q.F., Yin, F., Liu, C.L.: Improving handwritten Chinese text recognition by confidence transformation. In: 2011 International Conference on Document Analysis and Recognition, pp. 518–522. IEEE (2011)
23. Wang, S., Chen, L., Xu, L., Fan, W., Sun, J., Naoi, S.: Deep knowledge training and heterogeneous CNN for handwritten Chinese text recognition. In: Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition, pp. 84–89. IEEE (2016)
24. Wang, Z.X., Wang, Q.F., Yin, F., Liu, C.L.: Weakly supervised learning for oversegmentation based handwritten Chinese text recognition. In: Proceedings of the 17th International Conference on Frontiers in Handwriting Recognition, pp. 157–162. IEEE (2020)
25. Wu, L., et al.: R-drop: regularized dropout for neural networks. Adv. Neural Inf. Process. Syst. **34** (2021)
26. Wu, Y.C., Yin, F., Chen, Z., Liu, C.L.: Handwritten Chinese text recognition using separable multi-dimensional recurrent neural network. In: Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition, vol. 1, pp. 79–84. IEEE (2017)
27. Wu, Y.C., Yin, F., Liu, C.L.: Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models. Pattern Recogn. **65**, 251–264 (2017)
28. Xie, C., Lai, S., Liao, Q., Jin, L.: High Performance offline handwritten Chinese text recognition with a new data preprocessing and augmentation pipeline. In: Bai, X., Karatzas, D., Lopresti, D. (eds.) DAS 2020. LNCS, vol. 12116, pp. 45–59. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57058-3_4
29. Xie, Q., Dai, Z., Hovy, E., Luong, T., Le, Q.: Unsupervised data augmentation for consistency training. Adv. Neural. Inf. Process. Syst. **33**, 6256–6268 (2020)
30. Xie, Z., Huang, Y., Zhu, Y., Jin, L., Liu, Y., Xie, L.: Aggregation cross-entropy for sequence recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6538–6547 (2019)
31. Yang, H.M., Zhang, X.Y., Yin, F., Liu, C.L.: Robust classification with convolutional prototype learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3474–3482 (2018)
32. Yang, H.M., Zhang, X.Y., Yin, F., Yang, Q., Liu, C.L.: Convolutional prototype network for open set recognition. IEEE Trans. Pattern Anal. Mach. Intell. (2020)
33. Yin, F., Wang, Q.F., Zhang, X.Y., Liu, C.L.: ICDAR 2013 Chinese handwriting recognition competition. In: Proceedings of the 12th International Conference on Document Analysis and Recognition, pp. 1464–1470. IEEE (2013)
34. Yin, F., Wu, Y.C., Zhang, X.Y., Liu, C.L.: Scene text recognition with sliding convolutional character models. arXiv preprint arXiv:1709.01727 (2017)
35. Zhan, H., Wang, Q., Lu, Y.: Handwritten digit string recognition by combination of residual network and RNN-CTC. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.S. (eds.) ICONIP 2017. Lecture Notes in Computer Science, vol. 10639, pp. 583–591. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70136-3_62