LISTER: Neighbor Decoding for Length-Insensitive Scene Text Recognition

Changxu Cheng, Peng Wang, Cheng Da, Qi Zheng, Cong Yao
DAMO Academy, Alibaba Group
{ccx0127,wdp0072012,dc.dacheng08,zhengqisjtu,yaocong2010}@gmail.com

Abstract

*The diversity in length constitutes a significant characteristic of text. Due to the long-tail distribution of text lengths, most existing methods for scene text recognition (STR) only work well on short or seen-length text, lacking the capability of recognizing longer text or performing length extrapolation. This is a crucial issue, since the lengths of the text to be recognized are usually not given in advance in real-world applications, but it has not been adequately investigated in previous works. Therefore, we propose in this paper a method called **L**ength-**I**nsensitive **S**cene **TE**xt **R**ecognizer (LISTER), which remedies the limitation regarding the robustness to various text lengths. Specifically, a Neighbor Decoder is proposed to obtain accurate character attention maps with the assistance of a novel neighbor matrix regardless of the text lengths. Besides, a Feature Enhancement Module is devised to model the long-range dependency with low computation cost, which is able to perform iterations with the neighbor decoder to enhance the feature map progressively. To the best of our knowledge, we are the first to achieve effective length-insensitive scene text recognition. Extensive experiments demonstrate that the proposed LISTER algorithm exhibits obvious superiority on long text recognition and the ability for length extrapolation, while comparing favourably with the previous state-of-the-art methods on standard benchmarks for STR (mainly short text)*[1].

1. Introduction

Scene text recognition (STR) is a popular topic in the computer vision community [40, 41, 42, 34, 54, 18, 36, 48, 5], which aims at extracting machine-readable symbols from scene text images. Recently, a variety of works have pushed forward the recognition performance from the perspective of arbitrary-shaped text [42, 28, 57], combining language models [42, 54, 18, 36, 47], etc. However, sequence length, as a vital characteristic of text, is rarely discussed. In fact, instances of long text occur frequently in websites, compound words, text lines, codes and multi-lingual scenarios [7]. As revealed in previous studies [19, 49], existing methods cannot handle images with long text very well. We regard it as a key issue worthy of in-depth investigation. Concretely, we ought to systematically analyze the performance of existing methods on text of different lengths and explore an effective way to realize length-insensitive STR.

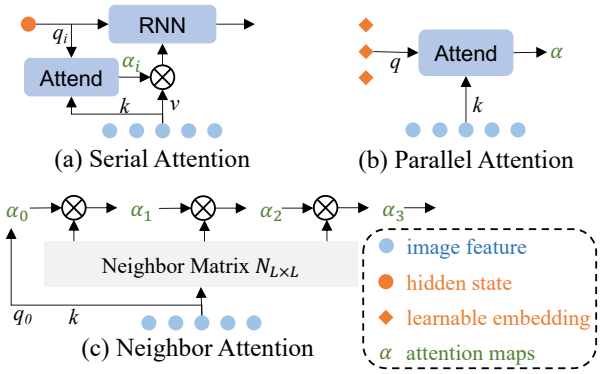


Figure 1. Different attention mechanisms for STR. α in green is the attention maps produced by: (a) complicated RNN in a serial way (Transformer decoder does similar), (b) pre-defined learnable limited queries in a parallel way, (c) the proposed novel neighbor matrix in a serial but simple and efficient way.

According to our observation, decoders in scene text recognition models are directly associated with the objective of text prediction, thus they are highly likely to play an important role in length-insensitive STR. In previous works, there are three types of text decoders: CTC [20, 40]-based decoder, serial attention decoder [42, 33, 5], and parallel attention decoder [49, 54, 18]. The CTC-based decoder makes dense prediction on the feature sequence, and then re-organizes the text by a pre-defined rule. It is efficient, robust to long text recognition, and can be applied to multi-line recognition [53, 50], but has some trouble in feature learning [30, 22]. The serial attention decoder (Fig. 1(a)) adopts RNN or the Transformer decoder to predict characters step by step, which is slow in inference, and may en-

[1] https://github.com/AlibabaResearch/AdvancedLiterateMachinery

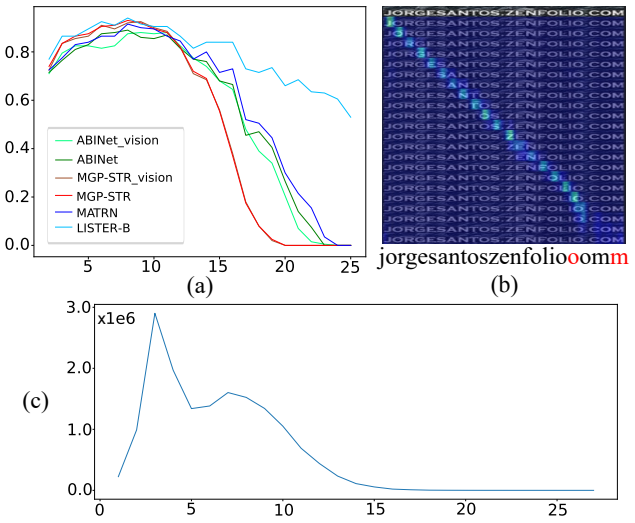


Figure 2. Performance of recent STR methods on images with different text lengths. (a) Accuracy distribution over length. (b) A case of attention map visualization of ABINet [18]. (c)Length distribution in the synthetic training set.

counter the attention drift problem [11, 49, 7], especially for long text. The parallel attention decoder (Fig. 1(b)) takes pre-defined limited queries to obtain character attention maps in a parallel way, which is efficient as CTC and effective for short text recognition. Thus, it has attracted considerable attention from the community recently. However, it appears to be sensitive to text length [19, 49]. In principle, each pre-defined query embedding requires a large amount of training data with the corresponding text lengths to learn well. Text of lengths unseen during training are difficult, if not impossible, to be well recognized.

To verify this, we simply evaluate the performance of several representative attention-based methods [18, 48, 36] on a scene text image dataset[2] where text lengths distribute uniformly. As shown in Fig. 2(a), the previous methods work very well for short text, but degrade dramatically as the text length becomes larger than 14. Fig. 2(b) shows the character attention maps where the latter maps are distracted, and the latter characters are also mis-recognized. Combining with the long-tailed length distribution of the training set [23, 21] (Fig. 2(c)), we conjecture that the parallel attention-based methods overfit on the text images of seen lengths seriously. In brief, the three existing decoders are not able to actualize effective length-insensitive STR.

On the other hand, the long-range feature modeling is crucial to the effectiveness of STR models. Recently, Transformer layers [44] have been proved strong for global context modeling [54, 18, 15, 48, 43, 5]. However, there are also concerns about the large computation cost [17, 5]. The computation complexity ($O(N^2)$) increases incontrovertible. Image patches or token features, including the noisy background, are all fed into the Transformer layers, which is a little bit redundant and expensive for GPU memory [58], especially for a large input size. Besides, the absolute positional encoding in Transformer layers also restricts the ability for length-insensitive STR.

Aware of the challenges above, we propose a **L**ength-**I**nsensitive **S**cene **TE**xt **R**ecognizer (LISTER), which incorporates a novel robust *Neighbor Decoder* (ND) and a lightweight *Feature Enhancement Module* (FEM). In ND, the attention mechanism is still utilized to ensure the effectiveness. Like the serial attention-based decoder, we regard the decoding process as a linked list data structure, so each character can be aligned by its previous neighbor regardless of its absolute position in the string. However, ND relies on a novel neighbor matrix where the next neighbor (character) locations of all the points in the feature map are indicated in a soft way, as shown in Fig. 1(c), which is simple but effective for length-insensitive text decoding. To model the long-range feature dependency with low computation cost, we propose to feed only the aligned character features to the Transformer layers and then enhance the whole feature map in FEM. The self-attention layers adopt the sliding window [6] to fit arbitrary-length sequences. Our contributions are summarized as follows:

1) A length-insensitive scene text recognizer, **LISTER**, is proposed with *neighbor decoder* as its core module. To the best of our knowledge, it is the first attempt to realize effective length-insensitive text recognition.

2) We propose a *Feature Enhancement Module* that enhance the whole feature map by Transformer layers with low computation cost.

3) Through extensive experiments, we prove that LISTER is on par with the previous state-of-the-arts on the common STR benchmarks, while outperforming them on long text. Besides, LISTER also achieves excellent performance in terms of length extrapolation.

2. Related Works

We review previous arts on scene text recognition (STR) based on the aspects related to our contributions [59, 32].

STR decoders. Text decoder is an essential component in a standard STR pipeline [2]. The CTC [20]-based decoder was adopted in various networks and applications [40, 31, 53, 50, 22, 16] due to its good balance between accuracy and efficiency. However, it was pointed out that the CTC loss misleads the feature alignments and representations[22]. The serial attention mechanisms in STR [42, 27, 11, 28, 9, 29, 33, 43, 5] were inspired by the attention-based encoder-decoder framework in machine

[2]The details of this dataset will be described in Sec. 4.1.

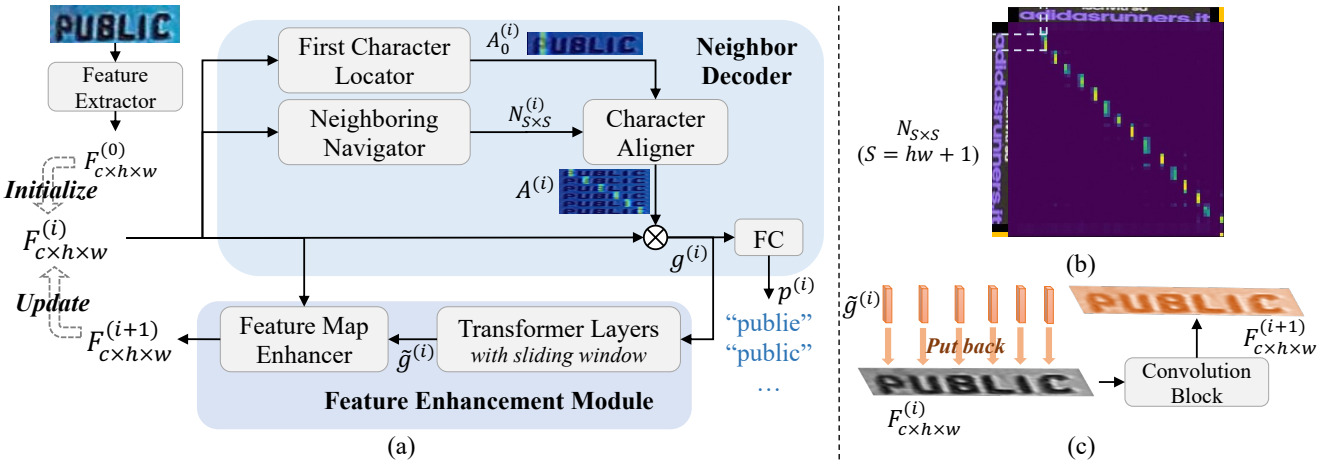


Figure 3. An overview of LISTER. (a) The model architecture. The Neighbor Decoder is the core component for length-insensitive STR, which takes $F^{(i)}$ as input and outputs string predictions $p^{(i)}$ and character features $g^{(i)}$. The Feature Enhancement Module enhances $F^{(i)}$ into $F^{(i+1)}$ with aid of $g^{(i)}$ with low computation cost. (b) The neighbor matrix generated by the Neighboring Navigator, which indicates the next neighbor of each character explicitly. (c) The feature map enhancer. The contextualized character features (in orange) are put back to $F^{(i)}$, then spread over $F^{(i)}$ by a convolution block.

translation [4] and the Transformer decoder [44]. Although the auto-regressive way achieved improved accuracy, the inference speed was also concerned since heavy computation remains in each decoding step. As a result, the parallel attention mechanism has become popular in recent years [49, 54, 46, 18, 38, 51, 48, 36]. It is efficient, but sensitive to text length for long text sequence [19]. It is urgent to have a robust, effective and efficient decoder.

Transformer in STR. The Transformer [44] architecture has been widely used in STR. Some works [54, 33, 18] used convolution layers and Transformer encoder layers as the image feature encoder. Recently, researchers built the backbone on the pure ViT [14, 1, 48, 43, 5] and achieved impressive performance. SVTR [15] used stacked local and global mixing blocks that contain shift-window attention and standard self-attention layers. Also, the Transformer decoder was directly taken as the text decoder in some arts [33, 43, 5]. However, it is non-trivial to reduce the significant computation cost caused by the Transformer layers. ABINet++ [17] replaced the Transformer units with stacked convolution layers directly. There are some works [24, 56] that proposed to use key frame proposals for efficient video understanding. Inspired by them, we only feed the key character features into the Transformer layers. Then the whole feature map is enhanced efficiently.

Problem of text length in STR. A key feature of the STR task is that text with arbitrary-length should be recognized. However, most prevoius arts [40, 42, 54, 18, 48] resized the input image to a fixed size and restricted the maximum length of text, which were unable to recognize unseen-length text absolutely. Meantime, for text with rarely-seen length, serial and parallel attention-based methods [42, 54, 18, 43, 36] are both more prone to make errors [49, 19]. Tan *et al*. [43] designed an ensemble model with patches of different resolutions to adapt to different degrees of stretching in fixed-size images. ABINet++ [17] took as input images with a fixed bigger size, aggregated the horizontal features, and enriched the query vectors with content to alleviate the multi-activation phenomenon for long text. In this work, we propose to build a text recognizer that is robust to text length.

3. Methodology

The proposed Length-Insensitive Scene Text Recognizer (LISTER) is able to read both short and long text images, and support iterative feature map enhancement. As shown in Fig. 3(a), LISTER consists of three parts: a feature extractor, a **Neighbor Decoder** (ND) and a **Feature Enhancement Module** (FEM). A text image is first fed to the feature extractor, then the encoded feature map $F^{(0)}_{c\times h\times w}$ is passed to the length-robust ND to obtain the character attention maps$A^{(0)}$, align the character features $g^{(0)}$ and get predictions $p^{(0)}$ (decoding iter 0). We further enhance the feature map by capturing the long-range dependency of the aligned character features in FEM. The enhanced feature map $F^{(i)}_{c\times h\times w}$ (i >0 means the feature map is enhanced i times) is also used in ND (decoding iter $1, \cdots$).

3.1. Feature Extractor

We adopt a convolution-based network, FocalNet [52], to extract visual features that does not acquire position encodings, which gets rid of the constraint of fixed image size. Similar to SVTR [15], we use strided depth-wise convolu-

tion and point-wise convolution to obtain image patch embeddings. Through the feature extractor, we obtain a feature map $F^{(0)}_{c\times h\times w}$ with rich local neighboring reception, where c is the number of feature channels.

3.2. Neighbor Decoder

The attention-based neighbor decoder is able to translate feature maps into arbitrary-length text robustly. Its attention mechanism consists of three simple modules, as shown in Fig. 3(a). We first flatten $F^{(i)}_{c\times h\times w}(i \geq 0)$ as $\tilde{F}^{(i)} \in \mathbb{R}^{(hw)\times c}$.

Neighboring Navigator navigates each feature point in $\tilde{F}^{(i)}$ where its *next neighbors* are. Here we demand that the next neighbors of a character-related point should be the points related to the next character. To this end, the neighbor matrix N is designed to indicate the next neighbors, as shown in Fig. 3(b): *N_{jk} is the probability of the k-th point being the next neighbor of the j-th*.

Note that we append an [EOS] token to $\tilde{F}^{(i)}$, so that the last character has the special token as its neighbor. The resulted full feature sequence is:

$$H^{(i)} = \left[\tilde{F}^{(i)}; e_{\text{EOS}}\right] \in \mathbb{R}^{S\times c} \tag{1}$$

where $[\cdot]$ is the concatenation operation, e_{EOS} is the learnable embedding of the [EOS] token, and $S = hw+1$. Then the neighbor matrix is obtained by a bilinear layer:

$$N^{(i)} = \sigma\left(\frac{1}{\sqrt{c}}\left(H^{(i)}W_q\right)W_r\left(H^{(i)}W_k\right)^{\mathrm{T}} + b_r\right) \tag{2}$$

where $N^{(i)} \in \mathbb{R}^{S\times S}$, W_q, W_k and W_r are learnable weights with shape $c\times c$, $b_r \in \mathbb{R}$, and σ is the softmax function.

First Character Locator generates the attention map of the first character. The query is a function of the global feature:

$$q^{(i)}_0 = \text{GAP}(F^{(i)})W_q \tag{3}$$

where $\text{GAP}(\cdot)$ is the global average pooling. Then the attention map of the first character is:

$$A^{(i)}_0 = \sigma\left(\frac{q^{(i)}_0}{\sqrt{c}}\left(H^{(i)}W_k\right)\right) \in \mathbb{R}^{1\times S} \tag{4}$$

Character Aligner produces all the character attention maps. Ideally, if the $(j-1)$-th character has been located at the J-th feature in $H^{(i)}$, the next character can be located by looking up the neighbor matrix $N^{(i)}$ based on the index J, which is $\text{argmax}N^{(i)}_J$. In real case, we implement the idea in a soft way. Given $N^{(i)}$ and $A^{(i)}_0$, $A^{(i)}_j(j>0)$ are calculated recurrently like a linked list:

$$A^{(i)}_j = A^{(i)}_{j-1}N^{(i)} \tag{5}$$

The attention map generation is going on until

$$A^{(i)}_{j,S-1} > \epsilon(0<\epsilon<1) \tag{6}$$

which means the end of decoding. ϵ is a hyper-parameter set to 0.6 empirically (the accuracy is insensitive to ϵ). Assuming that $L-1$ recurrences is performed, we obtain $A^{(i)} \in \mathbb{R}^{L\times S}$ which have $L-1$ real character attention maps and one [EOS] attention map. Since $\sum_k A^{(i)}_{0,k} = 1$, $\sum_k N^{(i)}_{lk} = 1$, we can easily deduce that $\sum_k A^{(i)}_{j,k} = 1$. Hence, $A^{(i)}$ is a matrix of probability distributions, which meets the demand of been taken as the character attention maps here.

Although Eq. (5) is performed in a serial way, it is still fast due to the low complexity of attention calculation. We will discuss it in the experiments. In fact, Eq. (5) is promising to run in parallel, since $A^{(i)}_j$ can also be calculated by only $A^{(i)}_0$ and $N^{(i)}$: $A^{(i)}_j = A^{(i)}_0 N^{(i)^j}$.

The aligned character features are:

$$g^{(i)} = A^{(i)}H^{(i)} \in \mathbb{R}^{L\times c} \tag{7}$$

Finally, we get the text prediction by a linear layer followed by a softmax function.

3.3. Feature Enhancement Module

Long-range feature modeling is important in scene text recognition. The proposed Feature Enhancement Module (FEM) captures the long-range dependency of the aligned character features, and then utilizes them to enhance the previous feature map $F^{(i)}$ into $F^{(i+1)}$ with low computation cost. We do not use language embedding to be elegant.

Specifically, the character features $g^{(i)}$ are fed into a few transformer layers $\mathcal{T}$ with sliding windows [6], to satisfy the original intention of processing arbitrary-length sequence:

$$\tilde{g}^{(i)} = \mathcal{T}(g^{(i)}) \in \mathbb{R}^{L\times c} \tag{8}$$

Fig. 3(c) illustrates the feature map enhancer. The contextualized character features $\tilde{g}^{(i)}$ are *put back* to the previous feature map, according to their attention maps:

$$\tilde{G}^{(i+1)} = H^{(i)} + A^{(i)\mathrm{T}}\tilde{g}^{(i)} \tag{9}$$

To further spread the long-range contextual information over the entire feature map, $\tilde{G}^{(i+1)}$ is reshaped to the size $h\times w\times c$ (the [EOS] feature is dropped), and a convolution block is simply exerted:

$$F^{(i+1)} = \mathcal{C}\left(\tilde{G}^{(i+1)}\right) \in \mathbb{R}^{c\times h\times w} \tag{10}$$

The enhanced feature map $F^{(i+1)}$ will also be decoded by the proposed neighbor decoder. The FEM+ND process can be performed iteratively to have finer and finer features.

3.4. Training and Evaluation

For each decoding iteration, the training objectives are three-fold as in Eq. (11), where λ_1 and λ_2 are set to 0.01 and 0.001 respectively. The recognition loss $\mathcal{L}_{\text{rec}}^{(i)}$ is the cross-entropy loss.

$$\mathcal{L}^{(i)} = \mathcal{L}_{\text{rec}}^{(i)} + \lambda_1 \mathcal{L}_{\text{eos}}^{(i)} + \lambda_2 \mathcal{L}_{\text{ent}}^{(i)} \tag{11}$$

To satisfy Eq. (6) at the end of decoding, we design an ending location loss as in Eq. (12), where $A_{L-1}^{(i)}$ is the attention map for the [EOS] token.

$$\mathcal{L}_{\text{eos}}^{(i)} = -\log A_{L-1,S-1}^{(i)} \tag{12}$$

Besides, to avoid the multi-activation phenomenon [17], the minimum entropy regularization is also used to make each attention map focus more on the core character region [10]:

$$\mathcal{L}_{\text{ent}}^{(i)} = -\frac{1}{L}\frac{1}{\log(1+S)} \sum_j \sum_k A_{jk}^{(i)} \log A_{jk}^{(i)} \tag{13}$$

For the model where FEM is iterated several times, the final objective takes the average loss.

During the inference, we propose an Attention Sharpening (AS) strategy to strengthen the ability for length extrapolation. Noted that an imprecise attention map may not lead to an prediction error for its corresponding character directly, but could influence the latter ones by attention error accumulation. Considering that an imprecise attention map usually has higher entropy, we propose to sharpen the character attention map when it is used for the next attention map generation. To this end, Eq. (5) is re-writen to:

$$A_j^{(i)} = \hat{A}_{j-1}^{(i)} N^{(i)} \tag{14}$$

$$\hat{A}_{j-1,s}^{(i)} = \frac{\exp\left(\alpha_j A_{j-1,s}^{(i)}\right) - 1}{\sum_t \exp\left(\alpha_j A_{j-1,t}^{(i)}\right) - S} \tag{15}$$

$$\alpha_j = \min\left(1 + \lambda(j-1), \mu\right) \tag{16}$$

where λ and μ are hyper-parameters set to 2 and 16 (insensitive) empirically. The resulted $\hat{A}_{j-1}^{(i)}$ is a sharpener distribution that could alleviate the accumulated attention errors. Finally, the prediction of the last iteration is adopted.

4. Experiments

4.1. Datasets

Common Benchmarks. We simply use the common benchmark datasets (CoB) following ABINet [18], to make fair comparison. The two widely-used synthetic datasets, MJSynth (MJ) [23] and SynthText (ST) [21], are used to train our model. Six real datasets are usually used to evaluate STR models, including 3 regular datasets (IIIT5K [35] (3000), IC13 [26] (857) and SVT [47] (647)) and 3 irregular datasets (IC15 [25] (1811), SVTP [37] (645) and CUTE [39] (288)).

Table 1. Architecture variants of LISTER.

Model	Depths	Dimensions	FEM
LISTER-T*	[2, 2, 6, 2]	[64, 128, 256, 512]	×
LISTER-T	[2, 2, 6, 2]	[64, 128, 256, 512]	✓
LISTER-B*	[2, 2, 9, 2]	[96, 192, 384, 768]	×
LISTER-B	[2, 2, 9, 2]	[96, 192, 384, 768]	✓

More challenging datasets. Following PARSeq [5], 3 challenging datasets are also taken for model evaluation: COCO-Text [45] (9.8k), ArT [12] (35.1k) and Uber-Text [55] (80.6k).

Based on the length distribution of the synthetic training set, text of length longer than 16 is assumed as long here. Long text images account for 0.3% in the 6 test sets of CoB, 1.1% in ArT, 0.5% in COCO-Text, and 1.9% in Uber-Text.

Text of Uniformly-distributed Lengths. The existing popular benchmarks do not have enough long text images to evaluate length-insensitive text recognition well. Hence, we collect a new scene text dataset as the new test set, named **Text of Uniformly-distributed Lengths** (TUL), where text of lengths 2-25 distributes uniformly, with 200 images and 200 different words for each length. To be clear, we only consider 36 characters here, including 26 English letters and 10 digits. The images are randomly sampled from the *Out of Vocabulary Scene Text Understanding* competition dataset[3] [19]. Images with very poor quality are filtered. We suggest that models evaluated on TUL should not be trained on real training set [5, 3], since there may be some overlaps between the real training data and TUL.

4.2. Implementation Details

For the feature extractor, the large receptive version of FocalNet [52] is used except that the image width is only 4x downsampled. For FEM, 8 heads are used in the self-attention layers with a window size of 11. Tab. 1 lists details of different LISTER variants. Throughout the experiments, we use the tiny version and 2 iterations of FEM (1 Transformer layer & 1 Convolution block) by default, unless some specifications. The number of classes is 37, including 26 lower-case letters, 10 digits and an end-of-sequence token.

For training, the model is initialized by using the truncated normal distribution. The AdamW optimizer is used with weight decay of 0.05, and the initial learning rate is $1e-3$ which starts with 5000 warm-up steps and decays to $5e-7$ by a cosine scheduler. The image augmentation

[3] https://rrc.cvc.uab.es/?ch=19&com=introduction

Table 2. Comparison with other methods. The results of some methods on TUL are evaluated based on their publicly-released models. LISTER-T*† is trained without $\mathcal{L}_{\text{ent}}^{(i)}$. LISTER-B$^{\#}$ adopts the 3-scale ensemble strategy during inference.

Method	Common Benchmarks (CoB)							TUL	Params (M)
	IIIT5K	IC13	SVT	IC15	SVTP	CUTE	AVG		
SRN [54]	94.8	95.5	91.5	82.7	85.1	87.8	90.4	-	54.7
ABINet$_{\text{vision}}$ [18]	94.6	94.9	90.4	81.7	84.2	86.5	89.8	56.1	23.5
VisionLAN [51]	95.8	95.7	91.7	83.7	86.0	88.5	91.2	-	32.8
SVTR-L [15]	96.3	97.2	91.7	86.6	88.4	**95.1**	92.8	-	40.8
MGP-STR$_{\text{vision}}$ [48]	96.4	96.5	93.2	86.3	89.5	90.6	92.7	50.6	85.5
PARSeq$_N$ [5]	95.7	96.3	92.6	85.1	87.9	91.4	92.0	-	23.8
PARSeq$_A$ [5]	**97.0**	97.0	93.6	86.5	88.9	92.2	93.2	-	23.8
PTIE [43]	96.3	97.2	94.9	**87.8**	90.1	91.7	**93.5**	-	45.9
ABINet [18]	96.2	97.4	93.5	86.0	89.3	89.2	92.6	57.7	36.7
MGP-STR [48]	96.4	97.3	94.7	87.2	**91.0**	90.3	93.3	50.8	148.0
MATRN [36]	96.6	**97.9**	**95.0**	86.6	90.6	93.5	**93.5**	60.5	44.2
LISTER-T*†	95.7	96.3	91.8	84.7	86.0	87.8	91.5	76.6	13.5
LISTER-T*	96.0	96.3	92.7	84.9	86.4	85.1	91.7	76.8	13.5
LISTER-B*	96.3	96.7	92.4	85.7	86.4	89.6	92.2	79.2	35.7
LISTER-T	96.5	97.7	93.5	86.5	87.8	87.9	92.8	77.0	19.9
LISTER-B	96.8	97.7	93.5	87.2	89.5	89.6	93.3	79.2	49.9
LISTER-B$^{\#}$	96.9	**97.9**	93.8	87.5	89.6	90.6	**93.5**	**79.5**	49.9

Table 3. Comparison on more challenging datasets.

Method	ArT	COCO	Uber	AVG
ABINet [18]	65.4	57.1	34.9	45.2
MATRN [36]	68.9	64.0	40.1	50.0
MGP-STR [48]	69.2	65.4	40.9	50.7
PARSeq$_N$ [5]	69.1	60.2	39.9	49.7
PARSeq$_A$ [5]	**70.7**	64.0	42.0	51.8
LISTER-T	69.0	64.1	48.0	55.0
LISTER-B	70.1	**65.8**	**49.0**	**56.2**
LISTER-B^{h}	**70.9**	**66.0**	**50.5**	**57.4**

technique in ABINet [18] is used. The height of images are set to 32, while the aspect ratios are randomly changed ranging from $\frac{1}{3}$ to 4 times. In the batch training, the batch width is the maximum width of the batch samples. Shorter images are filled with zero pads. Correspondingly, padding masks are exerted throughout the model. To be efficient, we empirically restrict the maximum width to 256 and 416 for different settings. There is no restriction on the maximum text length. We train our model for 10 epochs with a batch size of 512 using one A100-80GB card.

For both training and evaluation, images with the widths $w < 128$ are finally resized empirically[4] to avoid the side effect of some narrow text images. When comparing with PTIE [43] that proposed a strong ensemble way during evaluation, we also try a multi-scale ensemble strategy to improve the performance further. The input images are resized to 3 different scales[5], then fed to the model. The result with the highest probability is taken as the final prediction. It is very flexible and convenient since the input image size is unconstrained. we do not need to train an ensemble model.

[4] $h = 32, w' = w \times 0.33 + 85$

[5] The other 2 scaling ops: $w' = w \times 0.26 + 95$, $w' = w \times 0.21 + 121$

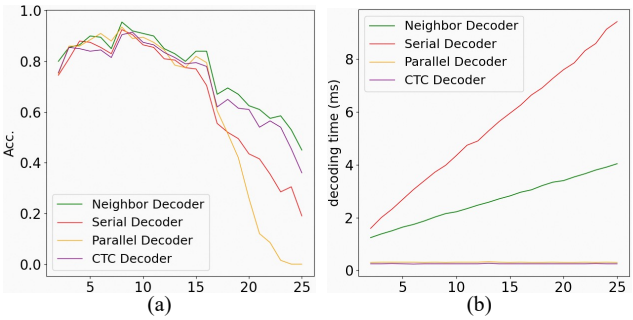
4.3. Comparison with State-of-the-arts

We compare our LISTER with other methods on TUL and the common benchmarks respectively.

Results on TUL. To compare the performance for length-insensitive text recognition, we evaluate several representative methods that are strong on the common benchmarks recently, on the collected TUL. As shown in Tab. 2, the total accuracies of them lag far behind the more lightweight LISTER-T* (improved by 16.3% at least). The main reason lies in that they are poor for long text images, which is seen clearly in Fig. 2. As the model size increases and FEM is used, LISTER-B gets a further improvement of 2.5%.

Results on Common Benchmarks. LISTER also does well on the common benchmarks where most texts are short. As shwon in Tab. 2, LISTER-T* gets 91.7% with only 13.5M parameters. Enhanced by FEM, LISTER-T gets 92.8% with 19.9 M parameters, reaching a good balance between accuracy and model size. By scaling up, LISTER-B achieves an accuracy of 93.3% that is nearly on par with the previous state-of-the-arts: MGP-STR [48] and MATRN [36] employed the external language priors to promote the final performance, while PTIE used the multi-resolution patch ensemble strategy. Through the simple 3-scale ensemble strategy (LISTER-B$^{\#}$), we obtain consistent gains, and achieves an average accuracy of 93.5%.

Results on Challenging Datasets. As shown in Tab. 3, LISTER-B outperforms others significantly on Uber-Text. One key factor may be the higher ratio of long text in Uber-



(a) (b)

Figure 4. Comparison of different decoders on TUL. (a) The accuracy distribution over different word lengths. (b) The decoding time distribution. (Best viewed in color.)

Table 4. Results of different decoders.

Decoder	CoB	TUL	MACs (G)	Time (ms)	Params (M)
CTC	90.2	72.9	**0.59**	**13.7**	**12.8**
PAT	91.4	61.6	0.60	13.8	13.0
SAT	91.3	66.5	0.73	18.1	15.4
ND	**91.7**	**76.8**	0.61	15.7	13.5

Text. However, LISTER-B does not get a better accuracy than PARSeq$_A$ on ArT where most cases have curved and rotated text. It is mainly owed to the fact that the height of extracted feature map is squeezed to 1 in our implementation, which is unfriendly to irregular-shape text. So we build another variant, LISTER-B^h, whose feature extractor outputs feature maps with $height = 4$. As can be seen, LISTER-B^h gets further improvement and catches the best.

4.4. Impact of Neighbor Decoder

The proposed neighbor decoder (ND) plays the key role for length-insensitive text recognition. We conduct experiments on several popular decoders to compare their performance comprehensively: CTC, parallel attention (PAT), serial attention (SAT) and ND. The implementation of PAT is based on SRN [54], and SAT is based on Aster [40] that does not need additional pre-defined positional query embeddings. To be fair, the experiments share all the settings except for the decoders, and SAT does not incorporate the internal language model. The speed is measured on TUL by one Nvidia V100 GPU card.

The results in Tab. 4 and Fig. 4 reveal that ND gets the best accuracies on both the common benchmarks and TUL with a little bit more cost. PAT does well on the common benchmarks where short text accounts for the most, but gets much worse on TUL. The accuracy decreases dramatically as the text length increases beyond 16. SAT has a slightly better accuracy on TUL with the cost of non-negligible MACs, latency and parameters. The attention-based RNN decoder still cannot adapt to long text well. CTC is poor at the common short text recognition, but more robust to text

wwwloudbillboardsccom (a) wwwloudbillboardscom (b)

Figure 5. Visualization of the attention maps for (a) parallel attention decoder and (b) the neighbor decoder. The character in yellow means it is missed, that in red means it is redundant or an error.

Table 5. Ablation study on the FEM module. TrFE denotes that the Transformer layers are placed in the end of the feature extractor.

TrFE	No. Iters	No. Trans	No. Conv	CoB	TUL	Params (M)	MACs (G)
✓	-	-	-	92.0	76.8	19.9	1.62
	0	-	-	91.7	76.8	13.5	0.61
	1	2	2	92.6	77.2	26.2	1.03
	2	0	1	92.0	76.5	16.7	0.85
	2	1	0	92.2	76.4	16.7	0.76
	2	1	1	**92.8**	77.0	19.9	1.05
	2	2	2	92.7	**77.8**	26.2	1.45

lengths, which owes to the essence that it conducts the decoding by dense predictions and re-ordering the characters by a general rule [20].

As for the decoding time, ND is slightly slower than CTC and PAT, but faster than SAT. Assuming that the growth rate of the time complexity $O(N)$ is τ, we find that τ of ND (0.122ms) is much less than that of SAT (0.340ms), as shown in Fig. 4(b). It is because the calculation in each attention generation step is very simple.

Character attention maps of a long text image are visualized in Fig. 5. Obviously, the attention maps of the latter characters are vague for PAT, which leads to missing, redundant or error character predictions. For ND, the attention map is focusing and clear, which is promising to obtain the exact character features.

4.5. Effects of Feature Enhancement Module

The proposed FEM module is responsible for the long-range dependency modeling with low computation cost. We study the effects of different iterations, the number of Transformer layers and convolution blocks, and where the Transformer layers should be placed.

From Tab. 5, we find that increasing iterations of FEM leads to better performance with the increasing computation cost (1/2/2 VS 2/2/2). The Transformer layer and the convolution block are both indispensable for FEM (by comparing

Input Image	w/o FEM	w/ FEM
	cuttage	cottage
	plzzza	pizza
	sall	saw

Figure 6. Some recognition cases. The character in red means it is redundant or an error.

Table 6. The results of the ability for length extrapolation on TUL.

Decoder	FEM	AS	Seen	Unseen	Total
CTC	-	-	82.9	56.2	72.9
ND	-	-	85.5	47.1	71.1
ND	-	✓	85.8	58.4	75.5
ND	✓	✓	**85.9**	**61.4**	**76.7**

2/0/1, 2/1/0 and 2/1/1). More Tansformer layers or convolution blocks contributes more on TUL (2/1/1 VS 2/2/2), which indicates that the long-range dependency in long text images is still important.

Some works [54, 18, 15] modeled the long-range dependency using Transformer layers in the latter part of their feature extractor. To explore the effects of them and ours fairly, we design a new setting: the Transformer layers are placed in the tail of the feature extractor too, without the FEM module. The first line in Tab. 5 shows that it does not work as well as FEM, but forwards with more computation cost. It may be because that too much noisy background features are also modeled in the Transformer layers, which is lack of the focusing on the key character features.

With iterations of FEM, the word prediction is refined. Surprisingly, FEM can edit the initial predicted wrong string by not only replacing characters, but also inserting and deleting. Fig. 6 shows some cases. This problem was pointed in SRN [54], and explored by LevOCR [13] interpretatively.

4.6. Ability for Length Extrapolation

An essential evidence for length-insensitive text recognition is the ability of length extrapolation, which means that the model should be able to recognize text of lengths unseen during training. To verify it, we filter the synthetic training set by a simple rule: images with text lengths > 16 are dropped. Then we train the models using the filtered training set. Note that we only train our LISTER and the CTC-based method for comparison. The PAT-based and SAT-based decoders are proven to perform poorly on long text images even under the training of full data in Fig. 4. The maximum width is increased to 416.

By the experiments, we conclude that ND have stronger ability for length extrapolation than CTC. As shown in Tab. 6 and Fig. 7, ND surpasses CTC on both seen- and unseen-length text. FEM brings a further improvement.

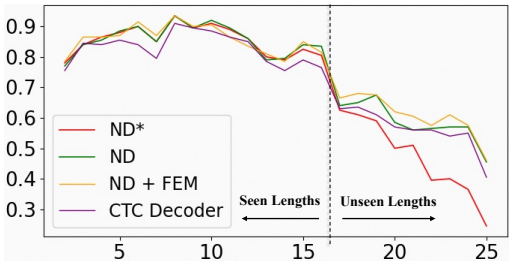


Figure 7. Results about length extrapolation. ND* is the Neighbor Decoder version without attention sharpening.

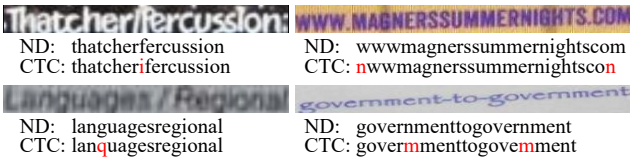


Figure 8. Recognition cases of unseen lengths in the length extrapolation study. CTC is prone to make errors on similar-shape characters or be interrupted by special characters. Note that we do not use FEM on these cases to be fair.

Figure 9. Visualization of (a) the neighbor matrix, and attention maps of (b) ND without AS and (c) ND with AS, for an unseen-length sample.

Some recognition cases are shown in Fig. 8.

A key factor for ND is the Attention Sharpening (AS) strategy. Without AS, ND does much worse than CTC on the unseen-length text. By visualization as in Fig. 9, we find that the neighbor matrix has no significant problems because the latter characters can still locate its next neighbor accurately. However, the latter attention maps of ND without AS (Fig. 9(b)) are vague, suffering a lot from the error accumulation. The little error in an early attention map will be enlarged as the attention map generation goes on. After using AS, the attention map (Fig. 9(c)) becomes clear and focusing, and the accuracy on the unseen-length text images increases by 11.3% notably.

4.7. Resizing or Padding?

Most previous works adopted direct resizing to preprocess the images. Some arts pointed out that the input images should be resized rather than padding in their framework [42, 43]. However, direct resizing causes image distortion inevitably [8], especially for the long text images.

For our framework, things have changed. As shown in

Table 7. Comparison on the ways of image preprocessing.

Decoder	Padding	Resizing	CoB	TUL
PAT	-	128	91.2	55.4
ND	-	128	91.2	58.7
ND	-	256	90.9	65.9
PAT	✓	-	91.4	61.6
ND	✓	-	**91.7**	**76.8**

Table 8. Comparison with other methods on scene-zh.

SAR [28]	**SRN** [54]	**SVTR-L** [15]	**LISTER-B**
62.5	60.1	72.1	**73.0**

Table 9. Results of different decoders on scene-zh.

Decoder	Short (59267)	Medium (3869)	Long (510)	Total (63646)
PAT	67.9	51.9	5.1	66.4
CTC	66.9	53.5	35.1	65.8
ND	**68.8**	**55.7**	**38.6**	**67.8**

Tab. 7, padding is always better for both PAT and ND. It may owe to the local-aware feature extractor that does not model the global dependency frequently like RNN or the global self-attention layer, so our feature extractor is not that easy to overfitting on the length.

4.8. Results on Chinese Text

Instances of long text may frequently occur in some non-Latin scenarios, such as Chinese. To demonstrate the superiority of ND in text images with real length distribution, we further conduct experiments on a Chinese scene text datase [7] (scene-zh), where a highly imbalanced length distribution prevails (1-77). The experiment settings follows Du *et al*. [15] to be fair, except that the maximum image width is extended to 448, and we do not restrict the maximum text length, which is an advantage of LISTER.

Our LISTER-B achieves state-of-the-art accuracy, as shown in Tab. 8. We also compare our ND with PAT and CTC on the short, medium and long text respectively. According to the length distribution, we manually regard texts of lengths < 12 as short, that > 25 as long, and others as medium. From Tab. 9, we conclude the same as discussed from Tab. 4, which confirms the strong ability of our length-insensitive text recognizer again, and the rationality of the collected TUL dataset. From another perspective, LISTER is promising to adapt to multi-lingual text recognition.

Although CTC decoder is also robust to text length, it has two problems revealed in Fig. 10. On the one hand, similar-shape characters are prone to be confused, as pointed in Fig. 8 as well. On the other hand, the congested successive same characters may be aggregated to only one due to the strict rule for character decoding (See the last case). Our ND is both robust to text length as CTC, and effective on character feature extraction.

Figure 10. Recognition cases in scene-zh. "_" in red means the character is missed.

5. Conclusion

We have presented a length-insensitive scene text recognizer, LISTER, which is robust to text length. As the core component, the attention-based neighbor decoder is able to obtain accurate attention maps through a novel neighbor matrix for text of arbitrary lengths robustly. To model the long-range dependency with low computation cost, we propose a Feature Enhancement Module where only the aligned features are fed to the Transformer layers. Extensive experiments have proved the effectiveness of LISTER on both short and long text images, as well as the capability of performing length extrapolation. In the future, we will explore a more efficient and robust way of image feature encoding for length-insensitive STR.

References

[1] Rowel Atienza. Vision transformer for fast and efficient scene text recognition. In *International Conference on Document Analysis and Recognition*, pages 319–334. Springer, 2021.

[2] Jeonghun Baek, Geewook Kim, Junyeop Lee, Sungrae Park, Dongyoon Han, Sangdoo Yun, Seong Joon Oh, and Hwalsuk Lee. What is wrong with scene text recognition model comparisons? dataset and model analysis. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4714–4722, 2019.

[3] Jeonghun Baek, Yusuke Matsui, and Kiyoharu Aizawa. What if we only use real datasets for scene text recognition? toward scene text recognition with fewer labels. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3112–3121, 2021.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[5] Darwin Bautista and Rowel Atienza. Scene text recognition with permuted autoregressive sequence models. *ArXiv*, abs/2207.06966, 2022.

[6] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150, 2020.

[7] Jingye Chen, Haiyang Yu, Jianqi Ma, Mengnan Guan, Xixi Xu, Xiaocong Wang, Shaobo Qu, Bin Li, and X. Xue. Benchmarking chinese text recognition: Datasets, baselines, and an empirical study. *ArXiv*, abs/2112.15093, 2021.

[8] Changxu Cheng, Qiuhui Huang, Xiang Bai, Bin Feng, and Wenyu Liu. Patch aggregator for scene text script identification. *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1077–1083, 2019.

[9] Changxu Cheng, Bohan Li, Qi Zheng, Yongpan Wang, and Wenyu Liu. Decoupling visual-semantic feature learning for robust scene text recognition. *ArXiv*, abs/2111.12351, 2021.

[10] Changxu Cheng, Wuheng Xu, Xiang Bai, Bin Feng, and Wenyu Liu. Maximum entropy regularization and chinese text recognition. In *Document Analysis Systems: 14th IAPR International Workshop, DAS 2020, Wuhan, China, July 26–29, 2020, Proceedings 14*, pages 3–17. Springer, 2020.

[11] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5086–5094, 2017.

[12] Chee Kheng Chng, Yuliang Liu, Yipeng Sun, Chun Chet Ng, Canjie Luo, Zihan Ni, ChuanMing Fang, Shuaitao Zhang, Junyu Han, Errui Ding, et al. Icdar2019 robust reading challenge on arbitrary-shaped text-rrc-art. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1571–1576. IEEE, 2019.

[13] Cheng Da, Peng Wang, and Cong Yao. Levenshtein OCR. In *European Conference on Computer Vision*, pages 322–338. Springer, 2022.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020.

[15] Yongkun Du, Zhineng Chen, Caiyan Jia, Xiaoyue Yin, Tianlun Zheng, Chenxia Li, Yuning Du, and Yu-Gang Jiang. Svtr: Scene text recognition with a single visual model. In *International Joint Conference on Artificial Intelligence*, 2022.

[16] Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Hongya Wang. Pp-ocr: A practical ultra lightweight ocr system. *ArXiv*, abs/2009.09941, 2020.

[17] Shancheng Fang, Zhendong Mao, Hongtao Xie, Yuxin Wang, Chenggang Clarence Yan, and Yongdong Zhang. Abinet++: Autonomous, bidirectional and iterative language modeling for scene text spotting. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2022.

[18] Shancheng Fang, Hongtao Xie, Yuxin Wang, Zhendong Mao, and Yongdong Zhang. Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7094–7103, 2021.

[19] Sergi Garcia-Bordils, Andrés Mafla, Ali Furkan Biten, Oren Nuriel, Aviad Aberdam, Shai Mazor, Ron Litman, and Dimosthenis Karatzas. Out-of-vocabulary challenge report. In *ECCV Workshops*, 2022.

[20] Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[21] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2315–2324, 2016.

[22] Wenyang Hu, Xiaocong Cai, Jun Hou, Shuai Yi, and Zhiping Lin. Gtc: Guided training of ctc towards efficient and accurate scene text recognition. In *AAAI Conference on Artificial Intelligence*, 2020.

[23] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *ArXiv*, abs/1406.2227, 2014.

[24] Zhengkai Jiang, Peng Gao, Chaoxu Guo, Qian Zhang, Shiming Xiang, and Chunhong Pan. Video object detection with locally-weighted deformable neighbors. In *AAAI Conference on Artificial Intelligence*, 2019.

[25] Dimosthenis Karatzas, Lluís Gómez i Bigorda, Anguelos Nicolaou, Suman K. Ghosh, Andrew D. Bagdanov, M. Iwamura, Jiri Matas, Lukás Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, Faisal Shafait, Seiichi Uchida, and Ernest Valveny. Icdar 2015 competition on robust reading. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160, 2015.

[26] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, M. Iwamura, Lluís Gómez i Bigorda, Sergi Robles Mestre, Joan Mas Romeu, David Fernández Mota, Jon Almazán, and Lluís-Pere de las Heras. Icdar 2013 robust reading competition. *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493, 2013.

[27] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2231–2239, 2016.

[28] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. *ArXiv*, abs/1811.00751, 2018.

[29] Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei A. F. Florêncio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. *ArXiv*, abs/2109.10282, 2021.

[30] Hu Liu, Sheng Jin, and Changshui Zhang. Connectionist temporal classification with maximum entropy regularization. In *Neural Information Processing Systems*, 2018.

[31] W. Liu, Chaofeng Chen, Kwan-Yee Kenneth Wong, Zhizhong Su, and Junyu Han. Star-net: A spatial attention residue network for scene text recognition. In *British Machine Vision Conference*, 2016.

[32] Shangbang Long, Xin He, and Cong Yao. Scene text detection and recognition: The deep learning era. *International Journal of Computer Vision*, 129:161–184, 2021.

[33] Ning Lu, Wenwen Yu, Xianbiao Qi, Yihao Chen, Ping Gong, and Rong Xiao. Master: Multi-aspect non-local network for scene text recognition. *Pattern Recognit.*, 117:107980, 2019.

[34] Pengyuan Lyu, Minghui Liao, Cong Yao, Wenhao Wu, and Xiang Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:532–548, 2018.

[35] Anand Mishra, Alahari Karteek, and C. V. Jawahar. Scene text recognition using higher order language priors. In *British Machine Vision Conference*, 2009.

[36] Byeonghu Na, Yoonsik Kim, and Sungrae Park. Multimodal text recognition networks: Interactive enhancements between visual and semantic features. In *European Conference on Computer Vision*, 2022.

[37] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. Recognizing text with perspective distortion in natural scenes. *2013 IEEE International Conference on Computer Vision*, pages 569–576, 2013.

[38] Zhi Qiao, Yu Zhou, Jin Wei, Wei Wang, Yuanqing Zhang, Ning Jiang, Hongbin Wang, and Weiping Wang. Pimnet: A parallel, iterative and mimicking network for scene text recognition. *Proceedings of the 29th ACM International Conference on Multimedia*, 2021.

[39] Anhar Risnumawan, Palaiahnakote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *Expert Syst. Appl.*, 41:8027–8048, 2014.

[40] Baoguang Shi, Xiang Bai, and Cong 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*, 39:2298–2304, 2016.

[41] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4168–4176, 2016.

[42] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Aster: An attentional scene text recognizer with flexible rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:2035–2048, 2019.

[43] Yew Lee Tan, Adams Wai-Kin Kong, and Jung-Jae Kim. Pure transformer with integrated experts for scene text recognition. *ArXiv*, abs/2211.04963, 2022.

[44] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017.

[45] Andreas Veit, Tomas Matera, Lukás Neumann, Jiri Matas, and Serge J. Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *ArXiv*, abs/1601.07140, 2016.

[46] Zhaoyi Wan, Minghang He, Haoran Chen, Xiang Bai, and Cong Yao. Textscanner: Reading characters in order for robust scene text recognition. In *AAAI Conference on Artificial Intelligence*, 2019.

[47] Kai Wang, Boris Babenko, and Serge J. Belongie. End-to-end scene text recognition. *2011 International Conference on Computer Vision*, pages 1457–1464, 2011.

[48] Peng Wang, Cheng Da, and Cong Yao. Multi-granularity prediction for scene text recognition. In *European Conference on Computer Vision*, 2022.

[49] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Canjie Luo, Xiaoxue Chen, Y. Wu, Qianying Wang, and Mingxiang Cai. Decoupled attention network for text recognition. In *AAAI Conference on Artificial Intelligence*, 2019.

[50] Tianwei Wang, Yuanzhi Zhu, Lianwen Jin, Dezhi Peng, Zhe Li, Mengchao He, Yongpan Wang, and Canjie Luo. Implicit feature alignment: Learn to convert text recognizer to text spotter. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5969–5978, 2021.

[51] Yuxin Wang, Hongtao Xie, Shancheng Fang, Jing Wang, Shenggao Zhu, and Yongdong Zhang. From two to one: A new scene text recognizer with visual language modeling network. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14174–14183, 2021.

[52] Jianwei Yang, Chunyuan Li, Xiyang Dai, and Jianfeng Gao. Focal modulation networks. *Advances in Neural Information Processing Systems*, 2022.

[53] Mohamed Yousef and Tom E. Bishop. Origaminet: Weakly-supervised, segmentation-free, one-step, full page text recognition by learning to unfold. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14698–14707, 2020.

[54] Deli Yu, Xuan Li, Chengquan Zhang, Junyu Han, Jingtuo Liu, and Errui Ding. Towards accurate scene text recognition with semantic reasoning networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12110–12119, 2020.

[55] Ying Zhang, Lionel Gueguen, Ilya Zharkov, Peter Zhang, Keith Seifert, and Ben Kadlec. Uber-text: A large-scale dataset for optical character recognition from street-level imagery. In *SUNw: Scene Understanding Workshop-CVPR*, volume 2017, page 5, 2017.

[56] Yuexi Zhang, Yin Wang, Octavia I. Camps, and Mario Sznaier. Key frame proposal network for efficient pose estimation in videos. *ArXiv*, abs/2007.15217, 2020.

[57] Dajian Zhong, Shujing Lyu, Palaiahnakote Shivakumara, Bing Yin, Jiajia Wu, Umapada Pal, and Yue Lu. Sgbanet: Semantic gan and balanced attention network for arbitrarily oriented scene text recognition. In *European Conference on Computer Vision*, 2022.

[58] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wan Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *ArXiv*, abs/2012.07436, 2020.

[59] Yingying Zhu, Cong Yao, and Xiang Bai. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science*, 10:19–36, 2016.