On Recognizing Texts of Arbitrary Shapes with 2D Self-Attention

Junyeop Lee Sungrae Park Jeonghun Baek
Seong Joon Oh Seonghyeon Kim Hwalsuk Lee*

Clova AI Research, NAVER/LINE Corp
{junyeop.lee, sungrae.park, jh.baek, seonghyeon.kim, hwalsuk.lee}@navercorp.com
coallaoh@linecorp.com

Abstract

*Scene text recognition (STR) is the task of recognizing character sequences in natural scenes. While there have been great advances in STR methods, current methods which convert two-dimensional (2D) image to one-dimensional (1D) feature map still fail to recognize texts in arbitrary shapes, such as heavily curved, rotated or vertically aligned texts, which are abundant in daily life (*e.g. *restaurant signs, product labels, company logos, etc). This paper introduces an architecture to recognize texts of arbitrary shapes, named* Self-Attention Text Recognition Network (SATRN). *SATRN utilizes the self-attention mechanism, which is originally proposed to capture the dependency between word tokens in a sentence, to describe 2D spatial dependencies of characters in a scene text image. Exploiting the full-graph propagation of self-attention, SATRN can recognize texts with arbitrary arrangements and large inter-character spacing. As a result, our model outperforms all existing STR models by a large margin of 4.5 pp on average in "irregular text" benchmarks and also achieved state-of-the-art performance in two "regular text" benchmarks. We provide empirical analyses that illustrate the inner mechanisms and the extent to which the model is applicable (*e.g. *rotated and multi-line text). We will open-source the code.*[1]

1. Introduction

Scene text recognition (STR) addresses the following problem: given an image patch tightly containing text taken from natural scenes (*e.g.* license plates and posters on the street), what is the sequence of characters? [34, 19] Applications of deep neural networks have led to great improvements in the performance of STR models [24, 14, 31, 4, 17, 3]. They typically combine a convolutional neural network (CNN) feature extractor, designed for abstracting the input patch, and recurrent neural network (RNN) encoder responsible for capturing sequential dependency, with a subsequent RNN character sequence generator, responsible for character decoding and language modeling.

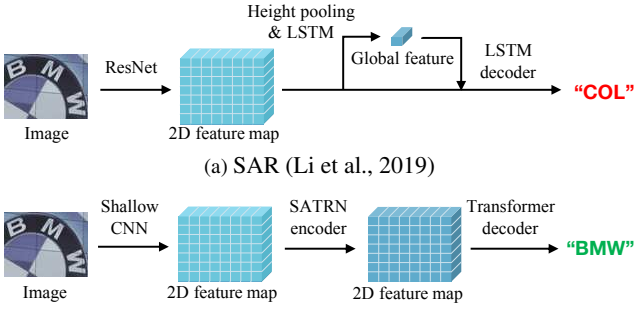


Figure 1: SATRN addresses the text images of difficult shapes (curved "BMW" logo) by adopting a self-attention mechanism, while keeping intermediate feature maps two dimensional. SATRN thus models long-range dependencies spanning 2D space, a feature necessary for recognizing texts of irregular geometry.

While these methods have brought advances in the field, they are built upon the assumption that input texts are written horizontally. Cheng *et al.* [4] and Shi *et al.* [24, 25], for example, have collapsed the height component of the 2D image into a 1D feature map. They are conceptually inept at interpreting texts with arbitrary shapes, which are important challenges in realistic deployment scenarios.

Realizing the significance and difficulty of recognizing texts of arbitrary shapes, the STR community has put more emphasis on such image types. The introduction of "irregular shape" STR benchmarks [2] is an evidence of such interest. On the method side, recent STR approaches are focusing more on addressing texts of irregular shapes. There are largely two lines of research: (1) input rectification and (2) usage of 2D feature maps. Input rectification [24, 25, 17, 18, 7] uses spatial transformer net-

*Corresponding author.
[1]https://github.com/clovaai/SATRN

works (STN, [12]) to normalize text images into canonical shapes: horizontally aligned characters of uniform heights and widths. These methods, however, suffer from the limitation that the possible family of transformations have to be specified beforehand and it is hard to normalize extreme case such as vertically aligned text.

Methods using 2D feature maps [5, 31, 15], on the other hand, extract 2D feature maps from input image without collapsing height component and sequentially retrieve characters on the 2D space. While the usage of 2D feature maps certainly increases room for more complex modelling, specific designs of existing methods are still limited by either the assumption that input texts are written horizontally (SAR [15]), overly complicated model structure (AON [5]), or requirement of ground truth character bounding boxes (ATR [31]). These methods also solely adopted Deep CNN based feature extractor, such as ResNet[9], to extract 2D feature maps from image, and there are no explicit design choices to capture the spatial dependency between characters in image, which is critical part in STR.

In this paper, we propose a STR network that adopts a 2D self-attention mechanism to capture the spatial dependency in 2D feature map to resolve the remaining challenging case within STR. Our architecture is inspired by the Transformer [27], which has made profound advances in the natural language processing [1, 6] and vision [21] fields. Our solution, *Self-Attention Text Recognition Network (SATRN)*, adopts the encoder-decoder construct of Transformer to address the cross-modality between the image input and the text output. The intermediate feature maps are two dimensional throughout the network. By never collapsing the height dimension, we better preserve the spatial information than prior approaches [15]. Figure 1 describes how SATRN preserves spatial information throughout the forward pass, unlike prior approaches.

Since the Transformer encoder is originally designed to capture the sequential dependency in 1D sequential input, there are several inappropriate aspects when it is adopted to 2D image. We propose a few simple modifications which is necessary to fully realize the benefit of self-attention in a 2D feature map. Three new modules are introduced: (1) Shallow CNN block, (2) Adaptive 2D positional encoding (A2DPE), and (3) Locality-aware feedforward layer (LAF). We will explain them in greater detail in the model section.

The resulting model, SATRN, is architecturally simple, memory efficient, and accurate. We have evaluated SATRN for its superior accuracy on the seven benchmark datasets and our newly introduced rotated and multi-line texts, along with its edge on computational cost. We note that SATRN is the state-of-the-art model in five out of seven benchmarks considered, with notable gain of 4.5 pp average boost on "irregular" benchmarks over the prior state-of-the-art.

Our contributions in this paper are threefold.

- We propose a network which adopts self-attention mechanism to resolve edge case within STR and achieves state-of-the-art performance in all "irregular" benchmark datasets.
- We propose useful modifications to make the Transformer encoder suitable for 2D input. And we also provide memory and speed analysis to demonstrate our models' superiority.
- We provide empirical analyses that illustrate how the self-attention works well in STR, as well as experiments on challenging cases, such as heavily rotated texts and multi-line text.

2. Related Works

In this section, we present prior works on scene text recognition, focusing on how they have attempted to address texts of arbitrary shapes. Then, we discuss previous works on using Transformer for visual tasks and compare how our approach differs from them.

2.1. Scene text recognition on arbitrary shapes

Early STR models have assumed texts are horizontally aligned. These methods have extracted width-directional 1D features from an input image and have transformed them into sequences of characters [24, 14, 31, 4, 17, 3, 22, 2]. By design, such models fail to address curved or rotated text. To overcome this issue, spatial transformation networks (STN) have been applied to align text image into a canonical shape (horizontal alignment and uniform character widths and heights) [24, 25, 17, 18, 7]. STN does handle non-canonical text shapes to some degree, but is limited by the hand-crafted design of transformation space and the loss in fine details due to image interpolation.

Instead of the input-level normalization, recent works have spread the normalization burden across multiple layers, by retaining two-dimensional feature maps up to certain layers in the network and information propagation across 2D space. Cheng et al. [5] have first computed four 1D features by projecting an intermediate 2D feature map in four directions. They have introduced a selection module to dynamically pick one of the four features. Their method is still confined to those four predefined directions. Yang et al. [31], on the other hand, have developed a 2D attention model over 2D features. The key disadvantage of their method is the need for expensive character-level supervision. Li et al. [15] have directly applied attention mechanism on 2D feature maps to generate text. However, their method loses full spatial information due to height pooling and RNN, thus being inherently biased towards horizontally aligned texts. These previous works have utilized a sequence generator sequentially attending to certain regions on the 2D feature map following the character order in texts.

In this work, we propose a simpler solution with the self-attention mechanism [27] applied on 2D feature maps. This approach enables character features to be aware of their spatial order and supports the sequence generator to track the order without any additional supervision.

2.2. Transformer for visual tasks

Transformer has been introduced in the natural language processing field [27, 6, 1]. By allowing long-range pairwise dependencies through self-attention, it has achieved breakthroughs in numerous benchmarks. The Transformer is a sequence-to-sequence model consisting of an encoder and decoder pair, without relying on any recurrent module.

Transformer has been adopted by methods solving general vision tasks such as action recognition [29], object detection [29], semantic segmentation [29, 10], and image generation [33, 21]. There also have been attempts to adopt a Transformer to a STR. They are, however, either limited to 1D self-attention [22] or applications of self-attention only on the decoder [20, 30]. We fully exploit the advantages of self-attention in the encoder on 2D feature maps.

3. SATRN Method

This section describes our model, *self-attention text recognition network (SATRN)*, in full detail. We will provide an overview of the SATRN architecture, and then focus on the newly introduced modules.

3.1. SATRN Overview

Figure 2 shows the overall architecture of SATRN. It consists of an encoder (left column), which embeds an image into a 2D feature map, and a decoder (right column), which then extracts a sequence of characters from the feature map.

Our contributions are focused on adapting the encoder to extract sequential information embedded in images along with arbitrary shapes. Meanwhile, most of the decoder modules are identical to the decoder of Transformer, and one can alternatively use the 2D LSTM decoder that is adopted in the previous method [15]. The effects of encoder and decoder are analyzed independently in 4.4.1.

3.1.1 Encoder

The encoder processes input image through a *Shallow CNN block* that captures local patterns and textures. The feature map is then passed to a stack of self-attention modules, together with an *Adaptive 2D positional encoding*, a novel positional encoding methodology developed for STR task. The self-attention modules are modified version of the original Transformer self-attention modules, where the point-wise feed forward is replaced by our *locality-aware feedforward layer*. The self-attention block is repeated N_e times

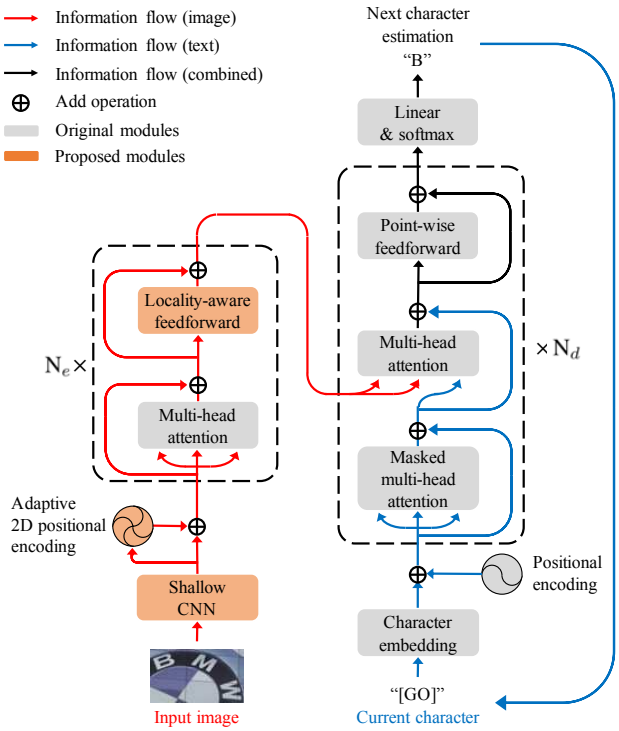


Figure 2: SATRN architecture overview. Left column is encoder and right column is decoder.

(without sharing weights). In the next section, we will describe in detail the components of SATRN that are newly introduced in the encoder.

3.1.2 Decoder

The decoder retrieves the enriched 2D features from the encoder to generate a sequence of characters. The cross-modality between image input and text output happens at the second multi-head attention module. The module retrieves the next character's visual feature. The feature of the current character is used to retrieve the next character's visual features upon the 2D feature map. Most of the decoder modules, such as multi-head attention and point-wise feedforward layers, are identical to the decoder of Transformer [27], as the decoder in our case also deals with sequence of characters [1].

3.2. Designing Encoder for STR

We explain how we have designed the encoder to effectively and efficiently extract sequential information from images. There are three useful modification on original self-attention block. Each of them will be explained.

3.2.1 Shallow CNN block

Input images are first processed through a shallow CNN. This stage extracts elementary patterns and textures in in-

put images for further processing in the subsequent self-attention blocks. Unlike in natural language processing, visual inputs tend to require much more abstraction as there are many background features to suppress (e.g. background texture of menu plate). Therefore, directly applying the self-attention block on the input image will be a great burden in terms of computational cost. This shallow CNN block performs pooling operations to reduce such a burden.

More specifically, the shallow CNN block consists of two convolution layers with 3×3 kernels, each followed by a max pooling layer with 2×2 kernel of stride 2.

3.2.2 Adaptive 2D positional encoding

The feature map produced by the shallow CNN is fed to self-attention blocks. The self-attention block, however, is agnostic to spatial arrangements of its input (just like a fully-connected layer). Therefore, the original Transformer has further injected positional information by adding *positional encoding(PE) vector*, which is embedded position values, to the 1D sequential feature maps.

PE has not been essential in vision tasks [33, 29, 10]; the focus in these cases has been to provide long-range dependencies not captured by convolutions. On the other hand, positional information plays an important role in recognizing text of arbitrary shape, since the self-attention itself is not supplied the absolute location information: given current character location exactly where in the image can we find the next character? Missing the positional information makes it hard for the model to sequentially track character positions. SATRN thus employs a 2D extension of the PE.

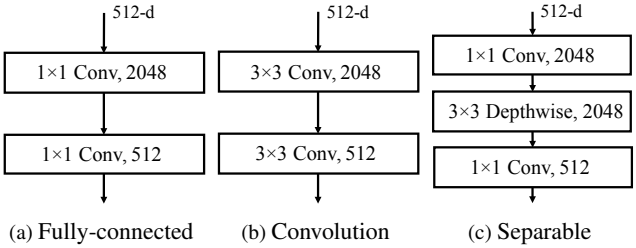
In STR, it is necessary to adaptively reflect the adjacency along the two directions according to the text alignment in image. For example, in the case of vertically aligned text, the adjacency of the height direction becomes more important factor than that of width direction in determining the order between characters. On the other hand, for horizontally aligned text, the adjacency in the width direction becomes more important. We thus propose the Adaptive 2D positional encoding (A2DPE) to dynamically determine the ratio between height PE and width PE element depending on the input image.

We first describe the self-attention module without PE. We denote the 2D feature maps produced by Shallow CNN block as $\mathbf{E}$ and its entry at position $(h, w) \in [1, ..., H] \times [1, ..., W]$ as $\mathbf{e}_{hw}$. The self-attention is computed as

$$\textbf{att-out}_{hw} = \sum_{h'w'} \text{softmax}(\text{rel}_{(h'w')\to(hw)})\mathbf{v}_{h'w'}, \quad (1)$$

where the value array $\mathbf{v}_{hw} = \mathbf{e}_{hw}\mathbf{W}^{\text{v}}$ is a transformation of the input feature through linear weights $\mathbf{W}^{\text{v}}$ and $\text{rel}_{(h'w')\to(hw)}$ is defined as

$$\text{rel}_{(h'w')\to(hw)} \propto \mathbf{e}_{hw}\mathbf{W}^{\text{q}}{\mathbf{W}^{\text{k}}}^{\text{T}}{\mathbf{e}_{h'w'}}^{\text{T}}, \quad (2)$$



(a) Fully-connected (b) Convolution (c) Separable

Figure 3: Locality-aware feedforward layer architecture options applied after the self-attention layer.

where $\mathbf{W}^{\text{q}}$ and $\mathbf{W}^{\text{k}}$ are linear weights that map the input into queries $\mathbf{q}_{hw} = \mathbf{e}_{hw}\mathbf{W}_q$ and keys $\mathbf{k}_{hw} = \mathbf{e}_{hw}\mathbf{W}_k$. Intuitively, $\text{rel}_{(h'w')\to(hw)}$ represents attention weights on feature at (h, w) when the query is feature at (h', w').

We now introduce our positional encoding A2DPE $\mathbf{p}_{hw}$ in this framework as below:

$$\text{rel}_{(h'w')\to(hw)} \propto (\mathbf{e}_{hw} + \mathbf{p}_{hw})\mathbf{W}^{\text{q}}{\mathbf{W}^{\text{k}}}^{\text{T}}(\mathbf{e}_{h'w'} + \mathbf{p}_{h'w'})^{\text{T}}. \quad (3)$$

Note that A2DPE are added on top of the input features. Now, A2DPE itself is defined as $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

$$\mathbf{p}_{hw} = \boldsymbol{\alpha}(\mathbf{E})\mathbf{p}_h^{\text{sinu}} + \boldsymbol{\beta}(\mathbf{E})\mathbf{p}_w^{\text{sinu}}, \quad (4)$$

where $\mathbf{p}_h^{\text{sinu}}$ and $\mathbf{p}_w^{\text{sinu}}$ are sinusoidal positional encoding over height and width, respectively, as defined in [27].

$$\mathbf{p}_{p,2i}^{\text{sinu}} = \sin(p/10000^{2i/D}), \quad (5)$$

$$\mathbf{p}_{p,2i+1}^{\text{sinu}} = \cos(p/10000^{2i/D}), \quad (6)$$

where p and i are indices along position and hidden dimensions, respectively. The scale factors, $\boldsymbol{\alpha}(\mathbf{E})$ and $\boldsymbol{\beta}(\mathbf{E})$, are computed from the input feature map $\mathbf{E}$ with 2-layer perceptron applied on global average pooled input feature as follows:

$$\boldsymbol{\alpha}(\mathbf{E}) = \text{sigmoid}\left(\max(0, \boldsymbol{g}(\mathbf{E})\mathbf{W}_1^{\text{h}})\mathbf{W}_2^{\text{h}}\right), \quad (7)$$

$$\boldsymbol{\beta}(\mathbf{E}) = \text{sigmoid}\left(\max(0, \boldsymbol{g}(\mathbf{E})\mathbf{W}_1^{\text{w}})\mathbf{W}_2^{\text{w}}\right), \quad (8)$$

where $\mathbf{W}_1^{\text{h}}$, $\mathbf{W}_2^{\text{h}}$, $\mathbf{W}_1^{\text{w}}$ and $\mathbf{W}_2^{\text{w}}$ are linear weights. The $g(\mathbf{E})$ indicates an average pooling over all features in $\mathbf{E}$. The outputs go through a sigmoid operation. The identified $\boldsymbol{\alpha}(\mathbf{E})$ and $\boldsymbol{\beta}(\mathbf{E})$ affects the height and width positional encoding directly to control the relative ratio between horizontal and vertical axes to express the spatial diversity. By learning to infer $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ from the input, A2DPE allows the model to adaptively reflect the adjacency of height and width directions when computing attention weights.

3.2.3 Locality-aware feedforward layer

For accurately recognizing characters in image, a model should not only utilize long-range dependencies but also lo-

cal vicinity around single characters. Thanks to the full-graph propagation characteristics, self-attention itself can capture long and short-term dependencies in the 2D feature map without any limitations, but it might be inefficient to exploit multi-stacked self-attention blocks to capture short-term dependency. We have thus improved the original point-wise feedforward layer (Figure 3a), consisting of two 1×1 convolutional layers by utilizing 3×3 convolutions (Figures 3b, 3c) to efficiently capture the short-term dependency in 2D feature map. In the experiments, we will show that between the naive 3×3 convolution and the depth-wise variant, the latter gives a better performance-efficiency trade-off.

4. Experiments

We report experimental results on our model. First, we evaluate the accuracy of our model against state-of-the-art methods on seven benchmark datasets. Second, we assess SATRN in terms of computational efficiency, namely memory consumption and the number of FLOPs and provide qualitative analysis of how our model can extract informative features from the input image. Third, we conduct ablation studies to evaluate our design choices including the Shallow CNN block, Adaptive 2D positional encoding, and the Locality-aware feedforward layer. Finally, we evaluate our model on more challenging cases not covered by current benchmarks, namely rotated and multi-lined texts.

4.1. STR Benchmark Datasets

Seven widely used real-word STR benchmark datasets are used for evaluation [2]. They are divided into two groups, "regular" and "irregular", according to the difficulty and geometric layout of texts.

Below are "regular" datasets that contain horizontally aligned texts. IIIT5K contains 3,000 images collected from the web, with mostly horizontal texts. Street View Text (SVT) consists of 647 images collected from the Google Street View. Many examples are severely corrupted by noise and blur. ICDAR2003 (IC03) contains 867 cropped text images taken in a mall. ICDAR2013 (IC13) consists of 1015 images inheriting most images from IC03.

"irregular" benchmarks contain more texts of arbitrary shapes. ICDAR2015 (IC15) consists of 2077 images that are taken in the wild without any specific prior action for improving its quality in the frame, which makes it more difficult to recognize text and closer to the real world problem. Street View Text Perspective (SVTP) consists of 645 images which text are typically captured in perspective views. CUTE80 (CT80) includes 288 heavily curved text images with high resolution. Samples are taken from the real world scenes in diverse domains.

4.2. Implementation Details

4.2.1 Training set

Two widely used training datasets for STR are Mjsynth and SynthText. Mjsynth is a 9M synthetic dataset for text recognition, generated by Jaderberg *et al*. [11]. SynthText represents 8M text boxes from 800K synthetic scene images, provided by Gupta *et al*. [8]. Most previous works have used these two synthetic datasets to learn diverse styles of synthetic sets, each generated with different engines. SATRN is trained on the combined training set, SynthText+Mjsynth, as suggested in Baek *et al*. [2] for fair comparison.

4.2.2 Architecture details

Input images are resized to 32×100 both during training and testing following common practice. The number of hidden units for self-attention layers is 512, and the number of filter units for feedforward layers is 4-times of the hidden unit. The number of self-attention layers in encoder and decoder are $N_e = 12$ and $N_d = 6$. The final output at each timestep is a vector of 94 scores; 10 for digits, 52 for alphabets, 31 for special characters, and 1 for the end token.

4.2.3 Optimization

Our model has been trained in an end-to-end manner using the cross-entropy loss. We have applied image rotation augmentation, where the amount of rotation follows the normal distribution $N(0, (34^\circ)^2)$. SATRN is trained with Adam optimizer [13] with the initial learning rate 3e-4. Cyclic learning rate [26] has been used, where the cycle step is 250,000. Batch size is 256, and the training is finished after 4 epochs. In our ablation study, we applied the same optimization strategy on baseline models for fair comparison.

4.2.4 Evaluation

We trained our model with spacial characters, adopting the suggestion by [2]. When we evaluate our model, we calculate the case-insensitive word accuracy [25]. Such training and evaluation method has been conducted in recent STR papers [25, 15, 16]. In our ablation studies, we use the unified evaluation dataset of all seven benchmarks (8,539 images in total) as done in [2].

4.3. Comparison against Prior STR Methods

We compare the performance of our model against existing STR models in Table 1. The accuracies for previous models are reported accuracies. Methods are grouped according to the dimensionality of feature maps, and whether the spatial transformer network (STN) has been used. The STN module and 2D feature maps have been designed to help recognizing texts of arbitrary shapes. We observe that

Method	Feature map	Training data	Regular test dataset				Irregular test dataset		
			IIIT5K	SVT	IC03	IC13	IC15	SVTP	CT80
CRNN [23]	1D	MJ	78.2	80.8	–	86.7	–	–	–
RARE [24]	1D	MJ	81.9	81.9	–	–	–	71.8	59.2
STAR-Net [18]	1D	MJ+PRI	83.3	83.6	–	89.1	–	73.5	–
GRCNN [28]	1D	MJ	80.8	81.5	–	–	–	–	–
FAN [4]	1D	MJ+ST+C	87.4	85.9	94.2	93.3	–	–	–
ASTER [25]	STN-1D	MJ+ST	**93.4**	**93.6**	–	91.8	76.1	78.5	79.5
Comb.Best [2]	STN-1D	MJ+ST	87.9	87.5	94.4	92.3	71.8	79.2	74.0
ESIR [32]	STN-1D	MJ+ST	93.3	90.2	–	–	76.9	79.6	83.3
ATR [31]	2D	PRI+C	–	–	–	–	–	75.8	69.3
AON [5]	2D	MJ+ST	87.0	82.8	91.5	–	68.2	73.0	76.8
CA-FCN [16]	2D	ST+C	92.0	82.1	–	91.4	–	–	79.9
SAR [15]	2D	MJ+ST	91.5	84.5	–	–	69.2	76.4	83.3
SATRN	2D	MJ+ST	92.8	91.3	**96.7**	**94.1**	**79.0**	**86.5**	**87.8**

Table 1: Scene text recognition accuracies (%) over seven benchmark test datasets. "Feature map" indicates the output shape of image encoder. "Regular" datasets consist of horizontally aligned texts and "irregular" datasets are made of more diverse text shapes. Accuracies of predicted sequences without dictionary matching are reported. In training data, MJ, ST, C and PRI denote MJSynth, SynthText, Character-labeled, and private data, respectively.

our model outperforms other 2D approaches on all benchmarks and that it attains the best performance on five of them against all prior methods considered. In particular, on irregular benchmarks that we aim to solve, SATRN improves upon the second best method [32] with a large margin of 4.5 pp on average.

4.4. Comparing SATRN against SAR

Since SATRN shares many similarities with SAR [15], where the difference is the choice of encoder (self-attention versus convolutions) and decoder (self-attention versus LSTM), we provide a more detailed analysis through comparison against SAR. We analyze the accuracy-efficiency trade-off as well as their qualitative differences.

4.4.1 Accuracy-efficiency trade-off

We analyze the contributions of encoder and decoder in our model, focusing both on the accuracy and efficiency. See Table 2 for ablative analysis. The baseline model is SAR [15] given in the first row (ResNet encoder with 2D attention LSTM decoder), and one can partially update SAR by replacing either only the encoder or the decoder of SATRN.

Encoder	Decoder	Params	FLOPs	Total
ResNet(2D)	LSTM	56M	21.9B	87.9
SATRN (2D)	LSTM	44M	16.4B	88.9
ResNet(2D)	SATRN	67M	41.4B	88.3
SATRN (2D)	SATRN	55M	35.9B	89.2

Table 2: Impact on accuracy and efficiency (the number of parameters and FLOPs) incurred by SATRN encoder and decoder. The first row corresponds to SAR [15] and the last is the proposed SATRN (ours). "Total" means word accuracy in unified seven benchmark datasets.

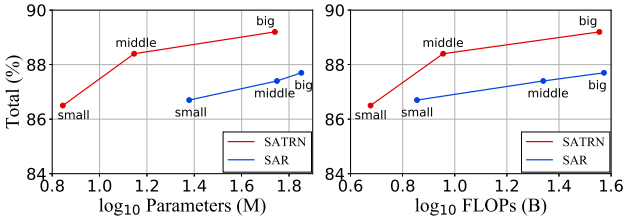


Figure 4: Accuracy-efficiency trade-off plots for SAR and SATRN. We have made variations, small, middle, and big, to control over the number of layers.

We observe that replacing ResNet encoder to SATRN encoder improve the accuracy by 1.0 pp and 0.9 pp over LSTM and SATRN decoders, respectively, while actually improving the space and time efficiency (reduction of 12M parameters and 5.5B FLOPs in both cases). This is the result of inherent computational efficiency enjoyed by self-attention layers and careful design of SATRN encoder to reduce FLOPs by modeling long-term and short-term dependencies of the features efficiently. This result shows that a self-attention based encoder can extract more informative feature maps from the input image compared to ResNet while reducing the number of parameters and FLOPs. The SATRN decoder, which is nearly identical to the original Transformer decoder, does provide further gain of 0.3 pp accuracy boost, but at the cost of increased memory consumption (+11M) and FLOPs (+19.5B).

To provide a broader view on the computational efficiency due to self-attention layers, we have made variations over SAR [15] and SATRN with varying number of lay-

(a) Character ROI (b) SA at depth 1 (c) SA at depth 2

Figure 5: Visualization of the self-attention maps.

ers. The original SAR contains ResNet34 as an encoder (SAR-middle), and we consider replacing the encoder with ResNet18 (SAR-small) and ResNet101 (SAR-big). Our base construct SATRN is considered SATRN-big which is identical with the model reported in Table 1. We consider reducing the channel dimensions in self-attention layers from 512 to 256 (SATRN-middle) and further reducing the number of encoder layers $N_e = 9$ and that of decoder layers $N_d = 3$ (SATRN-small).

Figure 4 compares the accuracy-efficiency trade-offs of SAR [15] and SATRN. We observe more clearly that SATRN design involving self-attention layers provides a better accuracy-efficiency trade-offs than SAR approach. We conclude that for addressing STR problems, our design is a favorable choice.

4.4.2 Qualitative comparison

We provide a qualitative analysis of how the 2D self-attention in encoder extract informative features from the input image. Figure 5 shows the human-defined character region of interest (ROI) as well as the corresponding self-attention heatmaps (SA) at depth n, generated by propagating the character ROI from the last layer to n layers below through self-attention weights. It shows the supporting signals relations at n for recognizing the designated character.

We observe that for character 'M' the last self-attention layer identifies the dependencies with the next character 'A'. SA at depth 2 already propagates the supporting signal globally, taking advantage of long-range connections in self-attention. By exploiting the long-range dependency, our model achieves high performance while removing redundancies created when stacking the deep convolution layer.

4.5. Ablation Studies on Proposed Modules

SATRN encoder is made of many design choices to adapt Transformer to the STR task. We report ablative studies on those factors in the following part, and experimentally analyze alternative design choices. The default model used hereafter is SATRN-small.

4.5.1 Adaptive 2D positional encoding (A2DPE)

This new positional encoding is necessary for dynamically adapting to overall text alignment (horizontal, diagonal, or vertical). As alternative options, we consider not doing any positional encoding at all ("None") [33, 29], using 1D positional encoding over flattened feature map ("1D-Flatten"), using concatenation of height and width positional encodings ("2D-Concat") [21]. In Table 3 we provides performance comparison between the proposed method and the baseline methods. In addition to the results on unified seven datasets, we also provide results on CUTE80 which contains heavily curved and irregularly aligned text images. The results show that A2DPE provides the best accuracy among four options considered and the improvement is more pronounced in the CUTE80.

Model	PE	CT80	Total
SATRN-small	None	73.6	83.8
SATRN-small	1D-Flat	78.5	85.8
SATRN-small	2D-Concat	80.2	85.8
SATRN-small	A2DPE	**81.3**	**86.5**

Table 3: Performance of SATRN-small with different positional encoding (PE) schemes.

(a) $r \in (0, 0.6)$ (b) $r \in (0.6, 0.8)$ (c) $r \in (0.8, \infty)$

Figure 6: Example of three groups of images separated by the weighting ratio of the height to width encoding vectors, $r = ||\boldsymbol{\alpha}(\mathbf{E})||_1 / ||\boldsymbol{\beta}(\mathbf{E})||_1$.

We provide additional visualization results to determine that A2DPE is working as we thought. Figure 6 is the result of dividing the images from benchmarks into three groups according to the weighting ratio of the height to width encoding vectors, $r = ||\boldsymbol{\alpha}(\mathbf{E})||_1 / ||\boldsymbol{\beta}(\mathbf{E})||_1$. As expected, Low ratio group(Figure 6a), which reflects the height encoding vector relatively less than width encoding vector, contains mostly horizontal samples, while high ratio group(Figure 6c) contains mostly vertical samples. By dynamically adjusting the reflection ratio of height encoding and width encoding, A2DPE reduces the representation burden for the other modules, leading to a performance boost.

4.5.2 Locality-aware feedforward layer

We proposed the locality-aware feedforward layers to capture the short-term dependency in 2D feature maps. To analyze their effects, we provide the performance comparison with the two alternatives described in Figure 3, with varying the number of encoder layers (3 or 9).

The resulting accuracy-performance trade-offs are shown in Table 4. Compared to the point-wise feedforward("FC"), naive convolution("Conv") results in improved

Model	N_e	Block	Params	FLOPs	Total
SATRN	3	FC	5M	3.90B	83.3
SATRN	3	Conv	18M	8.93B	85.7
SATRN	3	Separable	5M	3.91B	84.1
SATRN	9	FC	9M	5.57B	86.5
SATRN	9	Conv	47M	20.67B	88.3
SATRN	9	Separable	9M	5.60B	87.0

Table 4: Performance comparison of feedforward block according to the number of parameters and FLOPs.

Model	Rotated (IC13)				Multi-line
	0°	90°	180°	270°	
FAN (1D)	87.0	81.9	86.8	84.1	44.7
SAR (2D)	88.5	88.4	89.1	88.8	46.7
SATRN (2D)	**90.7**	**90.5**	**91.6**	**91.5**	**63.8**

Table 5: The results on two challenging text datasets; heavily rotated text and mutli-line text.

accuracy, but roughly with four times more parameters and FLOPs. We alleviate the computation cost with separable convolutions ("Separable") and achieve a better accuracy at nearly identical computational costs.

4.6. More Challenges: Rotated and Multi-Line Text

Irregular text recognition benchmarks (IC15, SVTP, and CT80) are attempts to shift the focus of STR research to more difficult challenges yet to be address by the field. While these datasets contain texts of more difficult shapes, it is not easy to analyze the impact of the *type* and *amount* of shape distortions. We have thus prepared new synthetic test sets (transformed from IC13) that consists purely of single type and degree of perturbation. Specifically, we measure the performance against texts with varying degrees of rotations (0°, 90°, 180°, and 270°) as well as multi-line texts.

We compare against two representative baseline models, FAN [4] and SAR [15]. Optimization and pre-processing details including training dataset and augmentation are unified for fair comparison.

4.6.1 Rotated text

Most STR models based upon the horizontal text assumption cannot handle heavily rotated texts. SATRN on the other hand does not rely on any such inductive bias; its ability to recognize rotated texts purely depends upon the ratio of such cases shown during training. To empirically validate this, we have trained the models with wider range of rotations: Uniform$(0°, 360°)$. Input images are then resized to 64×64. Second column group in Table 5 shows the results of rotated text experiments. We confirm that SATRN outperforms FAN and SAR while retaining stable performances for all rotation levels.

4.6.2 Multi-line text

We analyze the capability of models on recognizing multi-line texts, which would require the functionality to change line during inference. We have synthesized multi-line texts by concatenating SynthText and MJSynth along height dimension for training the models. For evaluation we have utilized multi-line text manually cropped from the scene images in IC13. Last column in Table 5 shows the results. SATRN indeed performs better than the baselines, showing its capability to make a long-range jump to change line during inference.

Figure 7 shows the attention map of the SATRN decoder to retrieve 2D features. SATRN distinguishes the two lines and successes to track the next line. The results show that SATRN enables the 2D attention transition from the current region to a non-adjacent region on the image.

Figure 7: The 2D attention maps on a multi-line example. The 2D attention follows the first text line and then moves to the next line.

5. Conclusions

Scene text recognition (STR) field has seen great advances in the last couple of years. Models are now working well on texts of canonical shapes. We argue that the important remaining challenge for STR is the recognition of texts with arbitrary shapes. To address this problem, we have proposed the *Self-Attention Text Recognition Network (SATRN)*. By allowing long-range dependencies through self-attention layer, SATRN is able to sequentially locate next characters even if they do not follow canonical arrangements. We have proposed several useful modules to adapt self-attention mechanism to STR task. We have achieved the new state of the art performances on irregular text recognition benchmarks with great margin (4.5 pp boost on average). SATRN has shown particularly good performance on our more controlled experiments on rotated and multi-line texts, ones that constitute the future STR challenges. We will open source the code.

References

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. *arXiv:1808.04444*, 2018. 2, 3

[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. In *2019 IEEE International Conference on Computer Vision*, 2019. 1, 2, 5, 6

[3] Fan Bai, Zhanzhan Cheng, Yi Niu, Shiliang Pu, and Shuigeng Zhou. Edit probability for scene text recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[4] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. In *2017 IEEE International Conference on Computer Vision*, pages 5086–5094, 2017. 1, 2, 6, 8

[5] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. AON: Towards arbitrarily-oriented text recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5571–5579, 2018. 2, 6

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019. 2, 3

[7] Yunze Gao, Yingying Chen, Jinqiao Wang, Zhen Lei, Xiao-Yu Zhang, and Hanqing Lu. Recurrent calibration network for irregular text recognition. *arXiv:1812.07145*, 2018. 1, 2

[8] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 5

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[10] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. CCNet: Criss-cross attention for semantic segmentation. In *2019 IEEE International Conference on Computer Vision*, 2019. 3, 4

[11] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. In *Workshop on Deep Learning, NIPS*, 2014. 5

[12] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 2

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015. 5

[14] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016. 1, 2

[15] Hui Li, Peng Wang, Chunhua Shen, and Guyu Zhang. Show, attend and read: A simple and strong baseline for irregular text recognition. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, 2019. 2, 3, 5, 6, 7, 8

[16] Minghui Liao, Jian Zhang, Zhaoyi Wan, Fengming Xie, Jiajun Liang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Scene text recognition from two-dimensional perspective. *arXiv:1809.06508*, 2018. 5, 6

[17] Wei Liu, Chaofeng Chen, and Kwan-Yee K. Wong. Char-net: A character-aware neural network for distorted scene text recognition. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, 2018. 1, 2

[18] Wei Liu, Chaofeng Chen, Kwan-Yee K. Wong, Zhizhong Su, and Junyu Han. Star-net: A spatial attention residue network for scene text recognition. In *Proceedings of the British Machine Vision Conference 2016, BMVC 2016*, 2016. 1, 2, 6

[19] Shangbang Long, Xin He, and Cong Ya. Scene text detection and recognition: The deep learning era. *arXiv:1811.04256*, 2018. 1

[20] Pengyuan Lyu, Zhicheng Yang, Xinhang Leng, Xiaojun Wu, Ruiyu Li, and Xiaoyong Shen. 2d attentional irregular scene text recognizer, 2019. 3

[21] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *arXiv:1802.05751*, 2018. 2, 3, 7

[22] Fenfen Sheng, Zhineng Chen, and Bo Xu. NRTR: A no-recurrence sequence-to-sequence model for scene text recognition. *arXiv:1806.00926*, 2018. 2, 3

[23] 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*, pages 2298–2304, 2017. 6

[24] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4168–4176, 2016. 1, 2, 6

[25] 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*, 2018. 1, 2, 5, 6

[26] Leslie N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 2017. 5

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 2, 3, 4

[28] Jianfeng Wang and Xiaolin Hu. Gated recurrent convolution neural network for ocr. In *Advances in Neural Information Processing Systems*, pages 334–343, 2017. 6

[29] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *CoRR*, abs/1711.07971, 2017. 3, 4, 7

[30] Lu Yang, Peng Wang, Hui Li, Ye Gao, Linjiang Zhang, Chunhua Shen, and Yanning Zhang. A simple and strong convolutional-attention network for irregular text recognition, 2019. 3

[31] Xiao Yang, Dafang He, Zihan Zhou, Daniel Kifer, and C Lee Giles. Learning to read irregular text with attention mecha-

nisms. In *26th International Joint Conference on Artificial Intelligence, IJCAI 2017*, 2017. 1, 2, 6

[32] Fangneng Zhan and Shijian Lu. ESIR: End-to-end scene text recognition via iterative image rectification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2059–2068, 2019. 6

[33] Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, 2019. 3, 4, 7

[34] Yingying Zhu, Cong Yao, and Xiang Bai. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science*, 10(1):19–36, 2016. 1