International Journal Of
Industrial Electronics Control and Optimization
Print ISSN: 3517-2645
Online ISSN: 3568-2645
Homepage: https://ieco.usb.ac.ir/

Investigation of Deep Learning Optimization Algorithms in Scene Text Detection

Zobeir Raisi [1,2] | John Zelek [1]

Systems Design Engineering Department, University of Waterloo, Waterloo, Ontario, Canada.[1]
Chabahar Maritime University, Chabahar, Iran.[2]
Corresponding author's email: zraisi@uwaterloo.ca

Article Info

Article type:
Research Article

Article history:
Received: 16-May-2023
Received in revised form: 01-Aug-2023
Accepted: 13-Aug-2023
Published online: 25-Aug-2023

Keywords:
Deep Learning,
Loss Function,
Optimization,
Scene Text Detection,
Attention Module

ABSTRACT

Scene text detection frameworks heavily rely on optimization methods for their successful operation. Choosing an appropriate optimizer is essential to performing recent scene text detection models. However, current deep learning methods often employ various optimization algorithms and loss functions without explicitly explaining their selections. This paper presents a simple segmentation-based text detection pipeline capable of handling arbitrary-shaped text instances in wild images. We explore the effectiveness of well-known deep-learning optimizers to enhance the pipeline's capabilities. Additionally, we introduce a novel Segmentation-based Attention Module (SAM) that enables the model to capture long-range dependencies of multi-scale feature maps and focus more accurately on regions likely to contain text instances. The performance of the proposed architecture is extensively evaluated through ablation experiments, exploring the impact of different optimization algorithms and the introduced SAM block. Furthermore, we compare the final model against state-of-the-art scene text detection techniques on three public benchmark datasets, namely ICDAR15, MSRA-TD500, and Total-Text. Our experimental results demonstrate that the Focal Loss (FL) combined with the Stochastic Gradient Descent (SGD) + Momentum optimizer using a poly learning-rate policy achieves a more robust and generalized detection performance than other optimization strategies. Moreover, our utilized architecture, empowered by the proposed SAM block, significantly enhances the overall detection performance, achieving competitive H-mean detection scores while maintaining superior efficiency in terms of Frames Per Second (FPS) compared to recent techniques. Our findings shed light on the importance of selecting appropriate optimization strategies and demonstrate the effectiveness of our proposed SAM in scene text detection tasks.

I. Introduction

Text Reading from wild images is a common research topic in computer vision with various valuable applications [1-4]. It has numerous practical applications such as optical character recognition, multi-lingual translation, image search, etc. [5-6]. Reading text from images in the environment can be broken down into two parts: (1) text detection, which focuses on locating the text within the image, and (2) text recognition, which involves converting the located text or individual word image into a string. This paper focuses on text detection, which is more complex than recognition due to wild images' variety of text shapes and complex backgrounds.

Before the deep learning era, text detection techniques relied on connected components or sliding window-based techniques,

which used handcrafted features such as MSER [7,8] or SWT [9]. These techniques had several significant areas for improvement: (1) they were designed only for detecting individual characters or components, making it challenging to identify regional context information and resulting in low recall performance. (2) They required multiple post-processing steps for text detection. (3) They were limited to horizontal text and could not handle multi-oriented text instances.

Recently, deep learning-based techniques are demonstrated superior performance to traditional classical machine learning-based methods for detecting various challenging text instances appearing in wild images. These deep-learning methods often utilize popular frameworks inherited from object detection communities, such as SSD [10], YOLO [11], Faster R-CNN [12], or object segmentation architectures such as FCN [13] and Mask R-CNN [14]. Deep learning-based text detectors mainly detect text at the word level, and they usually struggle with finding curved, very long, or abnormally shaped word instances by using only a single bounding box [15]. Improving the accuracy of text bounding boxes without significantly increasing the number of proposals, particularly in the case of curved or irregular-shaped text, is a crucial challenge in scene text detection.

Optimization plays a vital role in deep learning problems, and the objective is to find a way to optimize efficiency and determine the parameters that minimize the loss function [48, 49, 61]. Optimization algorithms [16-19] are defined by their update rule, which is regulated by hyper-parameters influencing its behavior (*e.g.*, the learning rate). However, no clear theory exists on which optimizer is best suited for scene text detection.

Attention, in the context of deep learning and natural language processing, refers to a mechanism that allows a model to focus on specific parts of the input data that are more relevant to the task. It enables the model to assign weights to different input parts, giving higher importance to the more informative elements. In computer vision, attention mechanisms are used successfully in tasks like image captioning, object detection, and visual question answering [50, 51]. In text detection, some recent methods [52, 53] benefited the attention mechanism in their models and achieved superior performance in benchmark datasets.

This paper utilizes a straightforward pipeline for scene text detection, which employs a ResNet [20] backbone for feature extraction followed by a Feature Pyramid Network (FPN) to obtain multi-scale feature maps. Furthermore, we design a specific attention module and insert it after FPN, namely the Segmentation-based Attention Module (SAM), to focus on different regions of the feature maps that are more likely to contain text. This framework makes the final model fast and capable of detecting arbitrary text shapes, such as curved or multi-oriented text. This paper also addresses the issue of various optimizers used in scene text detection architectures by examining the loss functions and optimization techniques commonly used in deep learning for object detection and segmentation. Moreover, it selects the best loss function and optimization method for the leveraged scene text detection framework.

Our main contributions are: (1) we comprehensively compare and analyze deep learning optimization algorithms to determine their effectiveness in text detection applications. (2) we design an attention-based module, SAM, to boost the detection performance of the utilized detector. (3) we conduct several ablation experiments to evaluate the effect of the designed block with the baseline architecture. (4) we compare the final proposed model with state-of-the-art text detection techniques on three well-known benchmark datasets.

II. Related Work

This section briefly overviews the deep learning-based techniques for scene text detection and examines the various loss functions and optimization methods used in these approaches.

A. Text Detection

Generally, scene text detection methods can be categorized as follows:

Regression-based (RB): Several recent deep learning-based scene text detection techniques [21, 22] used general object detection frameworks, like SSD [10] or Faster R-CNN [12]. These methods treat text regions as objects and predict candidate bounding boxes from them directly. For instance, TextBoxes [22] altered the SSD [10] kernels by using long default anchors and filters to handle the diverse aspect ratios of text instances and detect various text shapes. Unlike TextBoxes, the deep matching prior network (DMPNet) [23] employed sliding windows with quadrilateral shapes to catch text with multiple orientations.

Several regression-based methods [24, 25] attempted to tackle the detection challenges posed by rotated and arbitrarily shaped text. For example, EAST [21] introduced a fast and precise text detector that uses dense predictions and NMS to detect multi-oriented text in an image without needing manual anchors. Regression-based methods generally have a straightforward post-processing system for handling multi-oriented text. However, due to limitations in their structure, it is challenging to produce accurate bounding boxes for text instances with arbitrary shapes.

Segmentation-based (SB): Some techniques [26-28, 34] use pixel-level classification to detect text regions on a word or character level. These methods usually adopt a segmentation framework such as FCN [13] or Mask R-CNN [14]. For instance, Zhang et al. [26] applied FCN to generate a salience map of text regions. TextSnake [28] used FCN as a foundation and identified text instances by detecting and piecing together local parts.

The previously mentioned methods are designed to detect text in images at the word level. However, detecting text using word instances is difficult, as individual characters may have arbitrary shapes. As a result, some recent text detection techniques shifted towards character-level detection [15, 29, 30]. For instance, [29] generates multi-oriented text bounding boxes using a saliency map of text regions obtained from a segmentation network and character-level annotations. Another example is SegLink [30], which identifies small text segments in images and combines them to form word boxes through additional post-processing. CRAFT [15] uses a weakly supervised approach to detect individual characters in the arbitrarily shaped text, leading to state-of-the-art results in benchmark datasets. These recent methods, which adopt a segmentation framework, tend to better detect multi-oriented text than regression-based methods. However, they also come with more complex and time-consuming post-processing.

Several approaches [31, 32] aimed to simplify the architecture and enhance computation speed. Inspired by [32, 33], our method offers a streamlined pipeline that can accurately detect arbitrarily shaped text more efficiently than the various state-of-the-art methods [15, 21, 27, 28, 34, 52-60].

Attention-based Detectors: The attention mechanism helps the model to focus on the relevant parts of the image that are likely to contain text, improving the localization accuracy of the text detection system. By adapting the attention module to the specific characteristics of the text detection task, the model can achieve better results than the architecture without attention.

Many recent works [50-52] utilized attention in their pipeline to detect arbitrarily shaped text and object instances and achieved superior performance in benchmarks [50, 51]. For example, Hou et al. in [50] utilized coordinate information to selectively attend to relevant spatial locations in feature maps, improving model accuracy and reducing computational complexity. Tang et al. [52] presented a transformer-based approach for scene text detection, using representative features to avoid background disturbance and reduce computational costs. Their method achieves state-of-the-art results on popular datasets by effectively grouping components corresponding to text instances, enabling easy bounding box extraction without post-processing**.**

B. Loss Function for Deep Learning

The loss function represents our model's disagreement with the actual and predicted labels during the training phase. By evaluating the extent of this disagreement, we gain valuable insights into the model's learning progress and its ability to capture the underlying patterns within the data effectively. This section examines the standard loss functions utilized in recent deep learning-based scene text detection and object detection architectures.

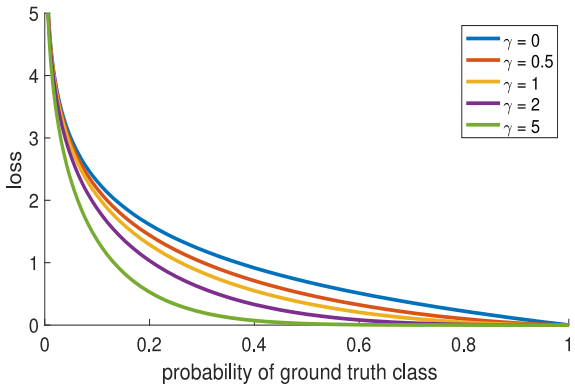

Fig. 1. Focal loss with different values of Gammas (γ).

Cross-Entropy (CE) Loss: This function evaluates the accuracy of a classification model, which gives an output between 0 and 1 in the form of probability. It is defined as:

$$\mathrm{CE}(p,y)=\begin{cases}-\log(p) & \text{if } y=1\\ -\log(1-p) & \text{otherwise.}\end{cases} \quad (1)$$

where $y \in \{\pm 1\}$ represents the ground-truth class and the $p \in [0,1]$ denotes the estimated probability of model for the class with label $y = 1$. For simplicity, we can define p_t as follows:

$$p_t=\begin{cases}p & \text{if } y=1\\ 1-p & \text{otherwise,}\end{cases} \quad (2)$$

Therefore, we can calculate $\mathrm{CE}(p,y)=\mathrm{CE}(p_t)=-\log(p_t)$. As evident from Figure 1, the cross-entropy loss has a drawback, shown in its plot (top), that it incurs a loss even for easily classified examples ($p_t > 0.5$) of non-trivial magnitude. When added over many straightforward examples, this could lead to small loss values, overpowering the rare class [35].

Focal Loss: The Focal Loss (FL), introduced in [35], is a modified version of the Cross-Entropy loss that assigns varying levels of importance to individual samples based on their classification error. The focal loss includes a modulation factor $(1-p_t)^{\gamma}$, controlled by the focusing parameter $\gamma \geq 0$, which is added to the Cross-Entropy loss. As shown in Figure 1, $\gamma = 0$ is utilized for Binay Cross Entropy (BCE) loss, $\gamma > 0$ is known as focal loss, and it outperforms BCE loss in object detection problems. The Focal Loss is defined as follows (more detail in Figure 1):

$$\mathrm{FL}(p_t)=-\alpha_t(1-p_t)^{\gamma}\log(p_t) \quad (3)$$

A balanced version of the Focal Loss is used in the implementation time with a default value of $\alpha = 0.9$. In this paper, we use $\gamma = 2$ in our experiments.

C. Optimization Algorithms for Deep Learning

The optimization algorithm minimizes or maximizes a given function, $f(x)$, in deep learning-based methods. Optimization aims to train the network by minimizing the loss function L, representing the difference between the predicted value y' and the actual value y. The predicted value y' is obtained

TABLE 1.
PARAMETERS AND THEIR SPECIFICATIONS THAT ARE USED IN OPTIMIZATION EQUATIONS.

Parameter	Specification
t	time step
w	weight
α	learning rate
$\partial\mathcal{L}/\partial w$	gradient of $\mathcal{L}$, Loss function, *w.r.t.* to w

through forward propagation, which uses the weights (w) and biases (b) of the network. Optimization algorithms update the values of W and b to minimize the cost function L. This section reviews some commonly used optimizers. A summary of the parameters and specifications used in these algorithms is provided in Table 1.

Given a function of $y = f(x)$, an optimization algorithm helps minimize or maximize the value of $f(x)$. In deep learning-based architectures, optimizers are used to train the frameworks by minimizing the loss function $\mathcal{L}$, which is defined as follows :

$$\mathcal{L}(w,b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_i\,(y_i', y_i) \tag{4}$$

The value of $\mathcal{L}$ is defined as the mean loss of the predicted value y' by the detector and the actual value y of the ground- truth.

Using the network's weights w and biases b during the forward propagation step of training, the values y' are computed. Furthermore, by updating the trainable parameters, w and b, we can minimize the value of the loss function $\mathcal{L}$ using different optimization algorithms. In the rest of this section, we review commonly used optimizers used in deep learning frameworks. Table 1 shows the commonly used parameters for explaining optimization algorithms.

Gradient Descent: is a first-order optimization algorithm used to minimize the cost function $\mathcal{L}$ and find the optimal values of the weight matrix w and bias b. It updates the parameters after going through the entire training data, which can be a challenge if the data is too large to fit in memory [36]. Mathematically it can be defined as:

$$w_{t+1} = w_t - \alpha\frac{\partial\mathcal{L}}{\partial w_t} \tag{5}$$

In Stochastic Gradient Descent (SGD), instead of evaluating the loss function and gradient using all the training examples, a small subset, known as a mini-batch (B), is randomly selected and used to compute an approximation of the total sum and the actual gradient in each iteration.

$$\frac{\partial\mathcal{L}(w)}{\partial w} = \frac{1}{B}\sum_{k=1}^{B}\frac{\partial\mathcal{L}_k(w)}{\partial(w)}, \tag{6}$$

$$w_{t+1} = w_t - \alpha\frac{\partial\mathcal{L}}{\partial w_t} \tag{7}$$

Mini-Batch Gradient Descent addresses the limitations of Gradient Descent by dividing the entire training data into smaller subsets, known as mini-batches, of sizes 8, 16, 32, or 64. These mini-batches are then used to train the network in a series of iterations. SGD offers several benefits compared to traditional Gradient Descent, such as more accurate data estimation, smoother convergence, a larger learning rate, and faster training. However, SGD also has some drawbacks, including the tendency to follow shallow dimensions in rapidly changing loss functions, getting stuck in local minima or saddle points (which are more common in high-dimensional problems like scene text detection), and the noise in gradient calculations from using mini-batches [19].

SGD+Momentum: To address the limitations of Stochastic Gradient Descent (SGD), adding a term called momentum [16] can be helpful. Thus we can define it as follows:

$$\begin{aligned} v_{t+1} &= \rho v_t + \frac{\partial\mathcal{L}}{\partial w_t} \\ w_{t+1} &= w_t - \alpha v_{t+1} \end{aligned} \tag{8}$$

where ρ denotes the friction constant; typically, its default value is 0.9 or 0.99. $v(t)$ is the velocity that builds up as a running mean of gradients.

AdaGrad: Adaptive gradient or AdaGrad [17], dynamically updates the learning rate at each update and for each weight individually, in which the gradient component remains unchanged like in SGD and is defined as follows:

$$\begin{aligned} w_{t+1} &= w_t - \frac{\alpha}{\sqrt{v_t+\epsilon}}\cdot\frac{\partial\mathcal{L}}{\partial w_t}, \\ v_t &= v_{t-1} + [\frac{\partial\mathcal{L}}{\partial w_t}]^2 \end{aligned} \tag{9}$$

In the above equation, ϵ shows a small positive value. However, the main weakness of AdaGrad is its learning rate, α, which is continuously decreasing and decaying. This can lead to slow or stalled training convergence, particularly for deep neural networks and complex models.

RMSprop: An alternative method for optimizing adaptive learning rates is RMSprop [18], designed to enhance AdaGrad [17]. Unlike AdaGrad [17], which accumulates the squared gradients, RMSprop employs an exponential moving average of these gradient values. RMSprop has now become a common choice, like momentum, for updating the learning rate component in the majority of optimization algorithms.

$$\begin{aligned} w_{t+1} &= w_t - \frac{\alpha}{\sqrt{v_t+\epsilon}}\cdot\frac{\partial\mathcal{L}}{\partial w_t}, \\ v_t &= \beta v_{t-1} + (1-\beta)[\frac{\partial\mathcal{L}}{\partial w_t}]^2 \end{aligned} \tag{10}$$

In the above equation, β is a hyperparameter with a default value of 0.9 during implementation as in [19].

Adam: Adaptive Moment (Adam) [19] is an optimization algorithm that merges the ideas of Momentum and RMSprop. It employs the squared gradients, like RMSprop, to adjust the

learning rate and incorporates momentum by utilizing the moving average of the gradient rather than the gradient itself, similar to SGD with momentum.

$$\begin{aligned} v_{t+1} &= \beta_1 v_t - (1-\beta_1)\frac{\partial \mathcal{L}}{\partial w_t}, \\ s_{t+1} &= \beta_2 s_t - (1-\beta_2)[\frac{\partial \mathcal{L}}{\partial w_t}]^2, \\ w_{t+1} &= w_t - \alpha \frac{v_{t+1}}{\sqrt{s_{t+1}+\epsilon}} \end{aligned} \quad (11)$$

The above β_1 and β_2 hyperparameters have a default value of 0.9 in practice [19].

All the optimization algorithms discussed above are commonly applied in scene text detection and recognition [36]. However, it is worth mentioning that other optimization algorithms also are used in deep learning object detection frameworks, such as Adadelta [37], Nadam [38], AdaMax [39], and AMSGrad [40], that are not usually used for scene text detection. We examine the suitability of each optimizer or learning rate for the scene text detection framework in Section IV.

III. Methodology

In this section, we introduce the baseline and proposed module of the segmentation-based pipeline for scene text detection. Figure 2 presents an overview of the entire architecture. The network utilizes a ResNet + Feature Pyramid Network (FPN) to extract multiscale feature maps from the input images (The left part of Figure 2). These features are subsequently processed by the proposed segmentation-based attention module (SAM), which enhances the model's ability to concentrate on regions in the image that likely contain text instances during the training phase (our main contribution in this paper, which is bonded with a yellow box in the middle). The obtained features are then employed to generate the text instance masks (The right part of Figure 2).

A. Text-detection Architecture

Feature Extraction: We leverage the ResNet-based backbone network (e.g., ResNet18 or ResNet50) [20] to extract features from the input image. This paper uses two light and deep backbones for feature extraction, ResNet18 and ResNet50. The ResNet18 is mainly utilized for ablation experiments, and the ReesNet-50 is used in our final model to fairly compare it with recent methods.

FPN: We then pass the extracted features through the FPN network [41] to obtain multi-scale feature maps (1/4, 1/8, 1/16}), enabling the text detectors to be more robust in detecting text of different sizes and scales in an image. The feature maps with different scales can be written as follows:

$$F = \{f_k \epsilon \mathbb{R}^{C \times H_k \times W_k} \mid k = 0,1,2\} \quad (12)$$

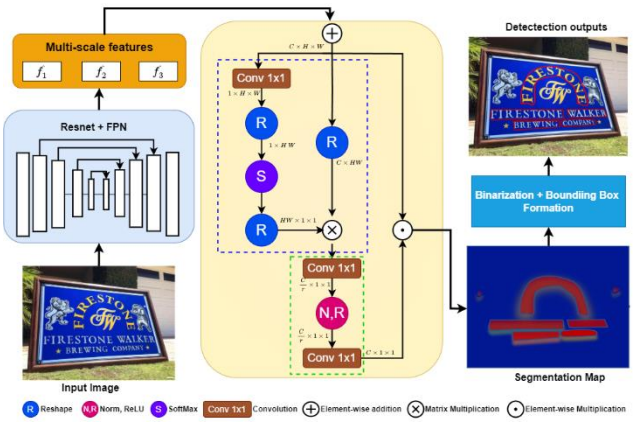

Fig. 2. The overall utilized scene text detection architecture.

The selected features are then concatenated (*concat*) and then fused by a 1x1 1D convolution layer (*conv*) as described as follows:

$$\hat{F} = Conv(Concat(f_k) \mid k = 0,1,2) \quad (13)$$

Segmentation-based Attention Module (SAM): We introduce an innovative attention module for the utilized segmentation-based architecture called SAM, as shown in Figure 2. Unlike [50, 51], SAM is designed to disseminate attention values across the spatial dimension coming from the FPN by using an element-wise multiplication strategy, which enables the model to focus more on different regions of the feature maps containing text instances.

Let $\hat{F}$ be the fused feature maps of equation (11); the output $\mathcal{S}$, which represents the global attention pooling for the SAM block (as shown with the blue dotted box in Figure 2), can be computed as follows:

$$\mathcal{S} = \hat{F}_{us} \otimes \left(\mathbb{c}_\gamma \sum_{\forall i} \frac{e^{\mathbb{c}_\alpha \hat{f}_i}}{\sum_{\forall j} e^{\mathbb{c}_\alpha \hat{f}_j}} \hat{F}_i \right) \quad (14)$$

where $\mathbb{c}_\gamma$ and $\mathbb{c}_\alpha$ are 1x1 convolutions that are learnable, $\hat{F}_{us}$ is a flattened version of vector $\hat{F}$. The j and i enumerate in the positions of all pixels. The symbol $\otimes$ represents the matrix multiplication.

Then, as depicted in the green dotted box of Figure 2, the variable $\mathbb{S}$ is transformed to capture channel-wise dependencies. The computation of this transformation can be expressed as:

$$\mathbb{S} = \sigma\left(\mathbb{c}_1\left(ReLU\left(LN(\mathbb{c}_2(\mathcal{S}))\right)\right)\right) \quad (15)$$

where σ shows the Sigmoid function. $LN(.)$ represents the Layer Normalization. $\mathbb{c}_1$ and $\mathbb{c}_2$ are 1×1 1D convolutions that are learnable.

Finally, using equations (13) and (15), the output of SAM, Γ , can be calculated as follows:

$$\Gamma = \mathbb{S} \odot \hat{F} \quad (16)$$

where $\odot$ is element-wise multiplication, the resulting outputs of equation (14) are then used to generate the segmentation map, as shown in Figure 2.

Binarization and Bounding Box Generation: After integrating the attention module into the baseline architecture, attended feature maps are obtained, as shown in Figure 2. These resulting features undergo post-processing steps to be binarized and highlight potential text regions. Using the binarized feature map, we employ differentiable binarization and bounding box label generation post-processing techniques, as described in [32, 33], to output the desired text detection. This enables the model to accurately identify and localize individual text regions within the image.

B. Loss Function

The loss function $\mathcal{L}$ is a combination of the loss for the probability map ($\mathcal{L}_s$) and the loss for the binary map ($\mathcal{L}_b$) and the loss for the threshold map $\mathcal{L}_t$ with weight factors α and β, which can be defined as follows:

$$\mathcal{L} = \mathcal{L}_{seg} + \alpha \times \mathcal{L}_b + \beta \times \mathcal{L}_t \tag{17}$$

where α and β are set to 0.9 during our experiments.

Previous scene text detection methods [21, 22, 26-28] faced challenges due to the extreme imbalance between foreground and background classes in training. Text regions occupy a small portion of the images, where the complex and straightforward examples are treated equally during training. We adopt the focal loss [35], modifying it to be more suitable for scene text detection, to address this imbalance issue. Therefore, the loss function can be defined as follows:

$$\begin{aligned}\mathcal{L}_{seg} = \mathcal{L}_b \quad &= \mathrm{FL}(\mathrm{Y}, \mathrm{Y}^*) = \frac{\sum_{i \in S_l} FL\,(p_t)}{|\mathrm{Y}^*|} \\ &= \frac{\sum_{i \in S_l} -\alpha_t (1-p_t)^{\gamma} \log p_t}{|\mathrm{Y}^*|}\end{aligned} \tag{18}$$

In (18), Y shows the score map prediction, and Y^* illustrates the ground-truth. S_l denotes sampled set where the ratio of positives and negatives is $1:3$. The γ parameter denotes the focusing parameter, and we set it to 2 in our experiments. This parameter reduces the loss contribution from easy samples ($p_t \to 1$) and balances the straightforward and complex examples on the pixel level during optimization. We show that the leveraged loss function performs better on convergence and improves the precision performance on segmentation tasks. Furthermore, for our experiments, we use the following Binary Cross Entropy loss to compare the results of it with focal loss:

$$\begin{aligned}\mathcal{L}_{seg} = \mathcal{L}_b = \sum_{i \in S_l} & Y_i \log Y_i^* \\ & + (1 - Y_i) \log (1 - Y_i^*)\end{aligned} \tag{19}$$

C. Optimization

Which Optimizer: Section II.B addressed the availability of numerous optimization methods for training deep learning models. However, choosing an appropriate optimizer or learning rate for a specific deep-learning framework remains to be determined. Despite the widespread use of adaptive optimization techniques, especially Adam, in scene text detection, their generalization and performance beyond the training set need to be better comprehended.

Learning Rate Policy: We use different learning rate policies like step decay or constant in our experiments while training our framework. However, we show that the "poly" learning rate policy, where the learning rate is multiplied by a specific equation, is more effective than the constant learning rate or "step" learning rate (which reduces the learning rate at a fixed step size).

$$\left(1 - \frac{\text{iter}}{\text{max\_iter}}\right)^{\text{power}} \tag{20}$$

We compare the effect of this learning rate with *"step"* and *"constant"* policy in Section IV.C and IV.D.

IV. Experimental Results and Discussion

In this section, first, we provide the implementation details of the models. Next, we briefly overview the utilized benchmark datasets and evaluation metrics. Then, we conduct multiple ablation studies of the baseline and proposed schemes. Finally, the proposed technique is tested and discussed quantitatively and qualitatively on three demanding public benchmarks: MSRA-TD500 [42], Total-Text [43], and ICDAR 2015 [44]. In the end, we discuss the limitation of the proposed model and offer insights into forthcoming efforts as future work to enhance its performance.

A. Implementation Details

We employ two models for the ablation experiments and comparison with state-of-the-art scene text detection methods. The *baseline* model utilizes a ResNet18 as the feature extractor, while the final model, used for comparison, is trained with the ResNet50 backbone. The final model (the *proposed model*) is pre-trained on 200K images from the Synthetic Text dataset [45] and subsequently fine-tuned until convergence on three real-world datasets (ICDAR15, Total-Text, MSRA-TD500 - one model for each dataset).

The parameters of the utilized loss functions are optimized using the SGD+ Momentum optimizer with a poly-learning policy. Training is conducted on 2 NVIDIA Tesla A100 GPUs, with a batch size of 4, and the entire process of training and fine-tuning takes approximately 30 hours. Various augmentation techniques, such as image resizing, random rotations, and horizontal flipping, are applied during training. All models are evaluated on a single GPU of NVIDIA RTX 3080Ti with 12GB of memory.

TABLE 2.
EFFECT OF DIFFERENT OPTIMIZATION METHODS ON SCENE TEXT DETECTION MODEL.

Optimizer	Iteration	Training Loss	Val H-mean
SGD	18K	2.83	29.11
	36K	2.42	29.25
	72K	2.11	32.11
SGD+Momentum	18K	2.65	32.97
	36K	2.35	33.45
	72K	**2.09**	**35.03**
Adam	18K	4.23	27.95
	36K	3.11	28.12
	72K	2.95	28.32

B. Datasets and Evaluation Metrics

SynthText: SynthText [45] in the wild is a large synthetic dataset of 858,750 artificial scene images used to pre-train our model. These images are created by combining natural images with text generated with different fonts, sizes, orientations, and colors. The annotations include both word and character-level rotated bounding boxes and text sequences. We use this synthetic dataset to pre-train our final model to compare with other frameworks.

ICDAR2015: The ICDAR2015 [44] dataset contains 1000 images for training and 500 for testing, with annotations at the word level represented by quadrilateral boxes. This dataset presents a higher difficulty level due to its text instances with varying orientation, illumination, and complex backgrounds, and most of the images are taken in indoor environments.

Total-Text: The TotalText dataset [43], introduced in ICDAR 2017, contains 1255 training and 300 testing images with text annotations provided as polygon shapes and word-level transcriptions.

MSRA-TD500: On the other hand, the MSRA-TD500 dataset [42] consists of 300 training images and 200 test images annotated with text lines. This dataset is challenging due to multi-lingual, arbitrary-oriented, and long text lines.

Evaluation Metric: In terms of quantitative evaluation, we use the ICDAR15 Intersection over Union (IoU) Metric [44] to measure the accuracy of the detection results.

$$IOU = \frac{A(G_j \cap D_i)}{Ar(G_j \cup D_i)} \tag{21}$$

Where $A(.)$ denotes area. The IoU is then calculated by comparing the jth ground-truth bounding box to the ith detection bounding box, and detection is considered correct if the IoU value is greater than or equal to 0.5. In addition, we use the H-mean, also known as the F-score, which is a combination of precision (P) and recall (R), to measure the performance of the pre-trained model of [15, 21, 27, 34] as described in [3] and the quantitative results of recent methods in [34,52-60].

$$\text{H-mean} = 2\frac{P \times R}{P + R} \tag{22}$$

TABLE 3.
EMPLOYING DIFFERENT LEARNING HYPERPARAMETERS ON THE TOTAL-TEXT [43] VALIDATION SET.

Learning policy	Batch size	Iteration	H-mean
constant	4	50K	60.36
step	4	50K	67.48
step	4	200K	75.35
poly	4	50K	70.23
poly	4	100K	71.81
poly	6	100K	72.58
poly	**4**	**200K**	**83.26**

For evaluating the test set of Total-Text and MSRA-TD500 datasets, we utilize similar evaluation metrics introduced in [43] and [42], respectively.

C. Effect Of Different Optimizers

To show the effect of the different optimizers, we first trained the baseline framework with ResNet18 backbone and without the SAM block on the 70% of SynthText dataset [47] for 80k iterations with a fixed learning rate of 1×10^{-4} in three SGD, SGD+Momentum (0.9), and Adam optimization algorithms. Adam works well in practice and outperforms other adaptive techniques, and it is the default optimizer in many deep learning-based detectors. Then, we evaluated our pre-trained models with three optimizers on the remaining 30% of the SynthText dataset as validation, as shown in Table 2. Although more epochs of training, we observed more decline in loss of the SGD optimizer compared to Adam in higher iterations.

Generalizability is a crucial characteristic in many deep learning models, which shows how a trained model on one dataset is capable of detecting challenging text in the validation set or other datasets; As can be seen from Table 2., SGD and SGD+Momentum provide better validation H-mean performance compared to Adam as the number of iteration increased. For instance, in the 72K iteration, SGD+Momentum achieved about 35%, while Adam remained at about 28%. Therefore, SGD has better generalization capability than Adam for our proposed method, and for the rest of this paper, we used SGD+Momentum to train our models.

D. Effect of Poly Learning Rate

Choosing a proper learning rate is also essential during the training of deep learning networks. For this purpose, we experimented with comparing the effect of different learning rates on our scene text detection problem. Table 3 illustrates the H-mean performance of employing "constant," "step," and "poly" learning rates. We used a fixed learning rate of 1×10^{-4} during training for constant policy. As shown in Table 3, employing "a poly" learning rate of (20) (with $power = 0.9$) and using the same batch size and same training

TABLE 4.
ABLATION STUDY TO EVALUATE THE EFFECT OF THE DESINED SAM BLOCK BY USING DIFFERENT BACKBONE AND LOSS FUNCTIONS ON THE TOTAL-TEXT [43] VALIDATION SET.

Model	Backbone	Attention module	Loss function	Iteration	H
Baseline	ResNet18	--	BCE	200K	79.3
Baseline	ResNet18	--	FL	200K	83.4
Proposed	ResNet18	SAM	FL	200K	84.9
Proposed	ResNet50	SAM	FL	200K	**86.5**

iterations yields 2.75% and 9.87% better performance than employing "step" and "constant" learning rate policy, respectively. The "constant" learning rate does not improve the H-mean performance of the model, and significantly has la ower performance in terms of H-mean from other learning rate policies.Therefore, we ignored it in the rest of our experiments. Fixing the batch size and increasing the training iteration to 100K improves the performance to 71.81% (1.58↑);

By increasing the batch size and the same iteration of 50K, we still experienced increased H-mean performance. We then reduced the batch size to 4 and found that comparable performance remains (71.81% versus 72.58%). Ultimately, we applied a batch size of 4 and 200K iterations to maintain similar training iterations as a "step" policy. We observed a significant H-mean performance of 83.26 (7.91% improvement over the "step" policy).

E. Effect of the Designed SAM, Feature Extraction Backbone, and Loss Functions

We conducted the following experiments on the benchmark Total-Text dataset to see the effect of the designed SAM block, different feature extraction backbones, and loss functions in the baseline and proposed model, shown in Table 4. We employed the "poly" learning policy and SGD+Momentom optimizer in all these experiments and set the batch size to 4.

We first eliminated the SAM and used the ResNet18 backbone to compare the effect of FL in (18) and BCE loss in (19). It can be seen from Table 4 that the model with focal loss outperformed the model with BCE loss using a similar setup of the model because this loss is more robust in challenging cases like the vertical or different scale of the text and designed to strike the imbalance issue of positive or negative text existing in the scene image. Then by applying the SAM to the architecture, as seen from Table 4, the H-mean performance boosted on a similar setup by ~1.5%, which confirms its effectiveness in the utilized architecture.

Recent methods in scene text detection [53-60] use ResNet50 instead of ResNet18 due to its deeper architecture and increased model capacity, enabling better feature representation and performance for handling complex and diverse visual patterns in text regions. We use the ResNet50 as our final feature extraction backbone to compare the recent models fairly. As shown, ResNet50 improved the detection performance of the model in terms of H-mean by ~1.5% compared to using ResNet18 because it allows the model to capture more complex and high-level features from the input images. The deeper architecture of ResNet50 provides a larger receptive field, which means that the network can consider a broader context of the input image.

TABLE 5.
QUANTITATIVE COMPARISON OF THE PROPOSED MODEL AMONG SOME OF THE RECENT METHODS ON TOTAL-TEXT DATASET.

Method	Total-Text			FPS
	Precision	Recall	H-mean	
CRAFT [15]	87.6	79.9	83.6	6.00
PSENet [34]	84.0	78.0	80.9	2.70
TextSnake [28]	82.7	74.5	78.4	1.10
DBNet [33]	87.1	82.5	84.7	19.0
DRRG [54]	86.5	84.9	85.7	3.50
FCENet [55]	89.3	82.5	85.8	–
PCR [56]	88.5	82.0	85.2	10.7
TextBPN [57]	90.3	84.7	87.4	10.6
TextDCT[53]	87.2	82.7	84.9	–
ABCNetV2[58]	89.2	84.1	87.0	10.0
FSG [52]	90.7	**85.7**	**88.1**	13.1
TPSNet [59]	89.2	85.0	86.6	11.6
LeafText [60]	**90.8**	84.0	87.3	–
Baseline	88.8	78.7	83.4	**31.0**
Proposed Model	90.5	82.9	86.5	**22.0**

F. Comparison with State-of-the-Art Text Detection Methods

We also compare our proposed model with several state-of-the-art scene text detectors [15, 28, 33, 34, 52-60] in Table 5 and Table 6 across three benchmark datasets: Total-Text, ICDAR15 [44], and MSRA-TD500 [42]. The proposed model utilizes a ResNet50 and the designed SAM block. The baseline model leverages a ResNet18 in its backbone as a feature extractor. Despite our final proposed model being pre-trained on a subset of synthetic dataset images and featuring a straightforward design, it achieved competitive performance in accuracy and efficiency, outperforming many recent state-of-the-art methods that often undergo pretraining on multiple datasets and involve complex design and post-processing.

The results in Table 5 demonstrate that the proposed model significantly outperforms the baseline model on the Total-Text dataset, which consists of many challenging text images, including curved and irregular text instances. The effectiveness of the SAM module is particularly evident in its recall performance, improving by approximately 4% compared to the baseline.

TABLE 6.
QUANTITATIVE COMPARISON OF THE PROPOSED MODEL AMONG SOME OF THE RECENT TEXT DETECTION METHODS ON ICDAR15 AND MSRA-TD500 DATASETS.

Method	ICDAR15			MSRA-TD500		
	Precision	Recall	H-mean	Precision	Recall	H-mean
EAST [21]	83.6	73.5	78.2	–	–	–
Pixellink [27]	82.8	81.6	82.2	83.0	73.2	77.8
CRAFT [15]	82.2	77.8	82.2	88.2	78.2	82.9
PSENet [34]	86.9	84.5	85.7	–	–	–
TextSnake [28]	84.9	80.4	82.6	83.2	73.9	78.3
DBNet [33]	86.8	78.4	82.3	91.5	79.2	84.9
DRRG [54]	88.5	84.7	86.6	88.0	82.3	85.0
FCENet [55]	90.1	82.6	86.2	–	–	–
PCR [56]	–	–	–	87.6	77.8	82.4
TextBPN [57]	–	–	–	86.6	84.5	85.6
TextDCT[53]	88.9	84.8	86.8	–	–	–
ABCNetV2[58]	90.4	86.0	88.1	89.4	81.3	85.2
FSG [52]	**90.9**	**87.3**	**89.1**	91.6	**84.8**	**88.1**
TPSNet [59]	90.5	85.1	87.7	–	–	–
LeafText [60]	88.9	82.3	86.1	**92.1**	83.8	87.8
Baseline	87.8	77.6	82.4	87.6	79.4	83.3
Proposed Model	90.3	81.3	85.6	89.8	82.4	85.9

As indicated in Table 5, the final model also demonstrated excellent performance in terms of Frames per second (FPS), achieving a rate of 22 FPS during inference, surpassing several other detectors. This high FPS capability highlights the applicability of our model in real-time detection applications. It is also worth mentioning that the designed SAM has minimal impact on the model's inference time.

As shown in Table 6, the designed SAM block and ResNet50 feature extractor boosted the performance of the proposed model by ~3% and ~2.5% in terms of H-mean performance on the ICDAR15 and MSRA-TD500 datasets, respectively, which are explicitly designed for multi-oriented scene text detection. The well performances on these datasets, confirm the proposed model's capability in detecting challenging text instances that are abundant in ICDAR15 and MSRA-TD500 datasets.

Finally, we present qualitative results in Figure 3, showcasing the detection performance of our proposed model on various samples from the Total-Text, ICDAR15, and MSRA-TD500 datasets. The results demonstrate the model's effectiveness in handling challenging cases, including multi-oriented, vertical, and curved text, small and complex fonts, and different languages, indicating its robustness and versatility across diverse scene text scenarios.

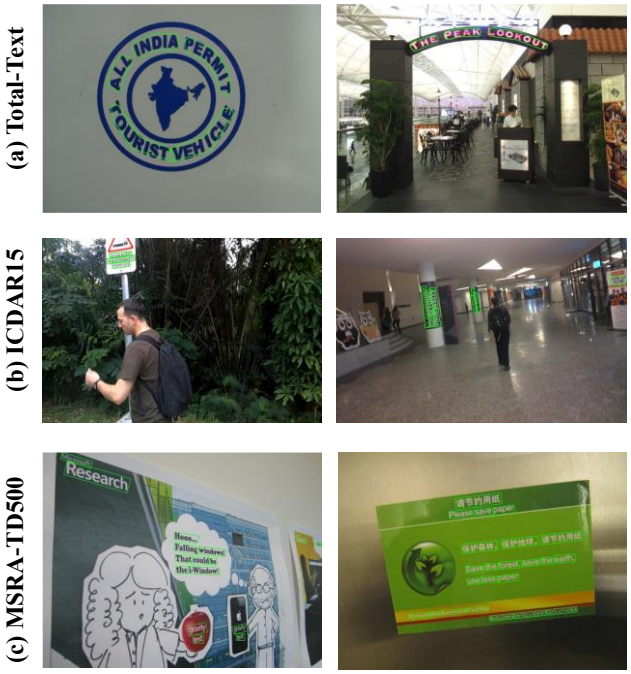

Fig. 3. Qualitative results of our final proposed method on benchmark datasets: (a) Total-Text [43], (b) ICDAR15 [44], and (c) MSRA-TD500 [42]. Detected text instances are shown with green color.

G. Limitation and Future Work for improvement

The presented model exhibits a significant limitation in cases where two-word instances in the image are closely positioned (i.e. when the first word's last character is close to the second word's first character). This scenario and the detection of crossed-word instances pose challenges and remain an unsolved problem in segmentation-based text detection architectures. Additionally, the proposed model encounters difficulty accurately detecting text instances under severe low resolution, low contrast, and occlusion conditions.

To address these limitations, the following suggested future work may enhance the model's ability to detect cha text instances. (1) One promising avenue for future work is to leverage recent advancements in natural language processing algorithms. Specifically, integrating pre-trained language model modules such as Generative Pre-training Transformer (GPT) [62] into the text detection and recognition framework could prove beneficial. By employing compositionality techniques [63, 64] alongside the language model, the model may be better equipped to infer and reconstruct the uncaptured characters in the detected text instances. (2) Another potential area for future research is exploring context-aware object interaction.

By incorporating contextual information from the surrounding objects and the overall scene, the model could better understand the text instances' spatial relationships and

textual coherence. This context-aware approach might enhance the model's ability to detect accurately and segment text instances closely intertwined or obscured by visual obstacles. (3) Given the challenges posed by low resolution, low contrast, and occlusion, another future direction is to explore adversarial training techniques. It could become more robust against various types of image degradation and perturbations by incorporating adversarial examples during the model's training process. This enhanced robustness might improve the model's performance in adverse real-world conditions, leading to more reliable text detection results.

V. Conclusions

This paper introduced a simple yet effective segmentation-based architecture for detecting text of arbitrary shapes in wild images. We thoroughly investigated deep-learning optimizers and loss functions applicable to our baseline pipeline. Additionally, we designed a specific attention block, the segmentation-based attention module (SAM), to focus on regions more likely to contain text instances. Through extensive ablation studies, we assessed the impact of the proposed module and analyzed different optimization algorithms and loss functions. Our experiments demonstrated that the SGD + Momentum optimizer with a "poly" learning rate policy offers superior detection performance, and the focal loss-based loss function proves more robust than binary cross-entropy loss. Comparing our model to recent state-of-the-art scene text detection methods on three benchmark datasets, our final model with the designed SAM significantly outperforms the baseline model and achieves competitive detection performance and efficiency. We also discussed the limitations of our architecture and outlined future research directions to further enhance text detection capabilities.

REFERENCES

[1] H. Lin, P. Yang, and F. Zhang, "Review of scene text detection and recognition, " *Archives of Computational Methods in Eng*., pp. 1–22, 2019.

[2] X. Liu, G. Meng, and C. Pan, "Scene text detection and recognition with advances in deep learning: A survey," *Int. J. on Document Anal. and Recognition (IJDAR)*, pp. 1–20, 2019.

[3] Z. Raisi, M. A. Naiel, G. Younes, P. Fieguth, and J. Zelek, "Smart text reader system for people who are blind using machine and deep learning, " *Machine Learning Algorithms for Signal and Image Processing*, pp. 161–200, 2022.

[4] Z. Raisi and J. Zelek, "Text detection & recognition in the wild for robot localization," *arXiv preprint arXiv:2205.08565*, 2022.

[5] Z. Raisi, M. A. Naiel, P. Fieguth, S. Wardell, and J. Zelek, "Text detection and recognition in the wild: A review, " arXiv preprint arXiv:2006.04305, 2020.

[6] Z. Raisi, "Text detection and recognition in the wild, " 2022. [7] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," Image and vision computing, vol. 22, no. 10, pp. 761–767, 2004.

[8] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images, " in Proceedings Asian Conference on Computer Vision. Springer, 2010, pp. 770–783.

[9] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2963–2970.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector, " in Eur. Conference on Computer Vision. Springer, 2016, pp. 21–37.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779– 788.

[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in Proceedings Adv. in Neural Info. Process. Sys., 2015, pp. 91–99.

[13] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.

[14] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in Proceedings IEEE Int. Conference on Computer Vision, 2017, pp. 2961–2969.

[15] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2019.

[16] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning, " in International conference on machine learning, 2013, pp. 1139– 1147.

[17] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," Journal of machine learning research, vol. 12, no. Jul, pp. 2121–2159, 2011.

[18] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," COURSERA: Neural networks for machine learning, vol. 4, no. 2, pp. 26–31, 2012.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition, " Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2015.

[21] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "East: an efficient and accurate scene text detector, " in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5551–5560.

[22] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network, " in Proceedings AAAI Conference on Artificial Intelligence, 2017.

[23] Y. Liu and L. Jin, "Deep matching prior network: Toward tighter multi-oriented text detection," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1962–1969.

[24] M. Liao, B. Shi, and X. Bai, "Textboxes++: A single-shot oriented scene text detector, " IEEE Trans. on Image Process. , vol. 27, no. 8, pp. 3676–3690, 2018.

[25] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals, "IEEE Trans. on Multimedia", vol. 20, no. 11, pp. 3111-3122, 2018.

[26] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4159–4167.

[27] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in Proceedings AAAI Conference on Artificial Intelligence, 2018.

[28] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "Textsnake: A flexible representation for detecting text of ar-arbitrary shapes," in Proceedings Eur. Conference on Computer Vision (ECCV), 2018, pp. 20–36.

[29] C. Yao, X. Bai, N. Sang, X. Zhou, S. Zhou, and Z. Cao, "Scene text detection via holistic, multi-channel prediction," arXiv preprint arXiv:1606.09002, 2016.

[30] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments, " in Proceedings *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2550–2558.

[31] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, and C. Shen, "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network, " in Proceedings of the IEEE Int. Conference on Computer Vision, 2019, pp. 8440–8449.

[32] P. Yang, G. Yang, X. Gong, P. Wu, X. Han, J. Wu, and C. Chen, "Instance segmentation network with self-distillation for scene text detection," IEEE Access, vol. 8, pp. 45 825–45 836, 2020.

[33] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai. Real-time scene text detection with differentiable binarization. In AAAI Conference on Artificial Intelligence, pages 11474–11481, 2020.

[34] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network, " arXiv preprint arXiv:1903.12473, 2019.

[35] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in Proceedings IEEE Int. Conference on Computer Vision, 2017, pp. 2980–2988.

[36] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning, " in Advances in Neural Information Processing Systems, 2017, pp. 4148–4158.

[37] M. D. Zeiler, "Adadelta: an adaptive learning rate method," arXiv preprint arXiv:1212.5701, 2012.

[38] T. Dozat, "Incorporating nesterov momentum into adam," 2016

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014

[40] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond, " arXiv preprint arXiv:1904.09237, 2019.

[41] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), July 2017.

[42] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images, " in IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012, pp. 1083–1090.

[43] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition, " in Proceedings IAPR Int. Conf. on Document Anal. and Recognition (ICDAR), vol. 1, 2017, pp. 935–942.

[44] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu et al., "Icdar 2015 competition on robust reading," in Proceedings Int. Conf. on Document Anal. and Recognition (ICDAR), 2015, pp. 1156–1160.

[45] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localization in natural images, " in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2016, last retrieved March 11, 2020.

[46] J. Liu, X. Liu, J. Sheng, D. Liang, X. Li, and Q. Liu, "Pyramid mask text detector, " CoRR, vol. abs/1903.11800, 2019.

[47] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localization in natural images, " in Proceedings IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2315–2324.

[48] N. Darjani and H. Omranpour, "Comprehensive learning polynomial auto-regressive model based on optimization with application of time series forecasting," International Journal of Industrial Electronics Control and Optimization, vol. 5,no. 1, pp. 43–50, 2022.

[49] S. Kalantari, M. Ramezani, and A. Madadi, "Introducing a new hybrid adaptive local optimal low-rank approximation method for denoising images," International Journal of Industrial Electronics Control and Optimization, vol. 3, no. 2, pp. 173–185, 2020.

[50] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 13 713–13 722.

[51] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in Proceedings of the IEEE/CVF international conference on Computer vision Workshops, 2019.

[52] J. Tang, W. Zhang, H. Liu, M. Yang, B. Jiang, G. Hu, and X. Bai, "Few could be better than all: Feature sampling and grouping for scene text detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 4563–4572.

[53] Y. Su, Z. Shao, Y. Zhou, F. Meng, H. Zhu, B. Liu, and R. Yao, "Textdct: Arbitrary-shaped text detection via discrete cosine transform mask," IEEE Transactions on Multimedia, 2022.

[54] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Liu, C. Yang, H. Wang, and X.-C. Yin, "Deep relational reasoning graph network for arbitrary shape text detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9699–9708.

[55] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, and W. Zhang, "Fourier contour embedding for arbitrary-shaped text detection," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 3123–3131.

[56] P. Dai, S. Zhang, H. Zhang, and X. Cao, "Progressive contour regression for arbitrary-shape scene text detection," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 7393–7402.

[57] S.-X. Zhang, X. Zhu, C. Yang, H. Wang, and X.-C. Yin, "Adaptive boundary proposal network for arbitrary shape text detection," in Proceedings of the IEEE/CVF International Conference on computer vision, 2021, pp. 1305–1314.

[58] Y. Liu, C. Shen, L. Jin, T. He, P. Chen, C. Liu, and H. Chen, "Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 11, pp. 8048–8064, 2021.

[59] W. Wang, Y. Zhou, J. Lv, D. Wu, G. Zhao, N. Jiang, and W. Wang, "Tpsnet: Reverse thinking of thin plate splines for arbitrary shape scene text representation," in Proceedings of the 30th ACM International Conference on Multimedia, 2022, pp. 5014–5025.
[60] C. Yang, M. Chen, Y. Yuan, and Q. Wang, "Text growing on leaf," IEEE Transactions on Multimedia, 2023.
[61] V. Nazarzehi and R. Damani, "Decentralised optimal deployment of mobile underwater sensors for covering layers of the ocean," Indonesian Journal of Electrical Engineering and Computer Science, vol. 25, no. 2, pp. 840–846, 2022.
[62] D. M. Katz, M. J. Bommarito, S. Gao, and P. Arredondo, "Gpt-4 passes the bar exam," Available at SSRN 4389233, 2023.
[63] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," Behavioral and brain sciences, vol. 40, p. e253, 2017.
[64] Z. Raisi and J. Zelek, "Occluded text detection and recognition in the wild," in 2022 19th Conference on Robots and Vision (CRV). IEEE, 2022, pp. 140–150.

Zobeir Raisi was born in Chabahar, Iran in 1987. He received his Ph.D. degree in 2022 from the Vision Image Processing Lab (VIPLab) at Systems Design Engineering Departments, University of Waterloo, Waterloo, Ontario, Canada. Currently, he is an assistant professor in the Department of Electrical Engineering, Chabahar Maritime University, Iran. His research interests include computer vision, artificial intelligence, robotics, and image processing.

John Zelek received his Ph.D. degree in philosophy of electrical engineering from the Centre for Intelligent Machines (CIM), McGill University, Montreal, QC, Canada, in 1996. He is currently a Professor of the Systems Design Engineering Department, University of Waterloo, Waterloo, ON, Canada and the Co-Director of the Vision Image Processing (VIP) Laboratory, University of Waterloo, ON, Canada. He has published over 300 refereed articles and has been a co-founder of five different startup companies from the University of Waterloo and has been an advisor for various other companies. His research interests include computer vision, AI, robotics, infrastructure monitoring, autonomous vehicles, image processing, augmented reality, assistive technology, to name a few.