PGNet: Real-time Arbitrarily-Shaped Text Spotting
with Point Gathering Network

Pengfei Wang,[1*] **Chengquan Zhang,**[2*] **Fei Qi,**[1*] **Shanshan Liu,**[2] **Xiaoqiang Zhang,**[2]
Pengyuan Lyu,[2] **Junyu Han,**[2] **Jingtuo Liu,**[2] **Errui Ding,**[2] **Guangming Shi**[1]

[1]School of Artificial Intelligence, Xidian University,
[2]Department of Computer Vision Technology, Baidu Inc.
pfwang@stu.xidian.edu.cn, zhangchengquan@baidu.com, fred.qi@ieee.org, {liushanshan07,
zhangxiaoqiang01}@baidu.com, lvpyuan@gmail.com, {hanjunyu, liujingtuo, dingerrui}@baidu.com, gmshi@xidian.edu.cn

Abstract

The reading of arbitrarily-shaped text has received increasing research attention. However, existing text spotters are mostly built on two-stage frameworks or character-based methods, which suffer from either Non-Maximum Suppression (NMS), Region-of-Interest (RoI) operations, or character-level annotations. In this paper, to address the above problems, we propose a novel fully convolutional Point Gathering Network (PGNet) for reading arbitrarily-shaped text in real-time. The PGNet is a single-shot text spotter, where the pixel-level character classification map is learned with proposed PG-CTC loss avoiding the usage of character-level annotations. With a PG-CTC decoder, we gather high-level character classification vectors from two-dimensional space and decode them into text symbols without NMS and RoI operations involved, which guarantees high efficiency. Additionally, reasoning the relations between each character and its neighbors, a graph refinement module (GRM) is proposed to optimize the coarse recognition and improve the end-to-end performance. Experiments prove that the proposed method achieves competitive accuracy, meanwhile significantly improving the running speed. In particular, in Total-Text, it runs at 46.7 FPS, surpassing the previous spotters with a large margin.

1 Introduction

Recently, scene text reading has attracted extensive attention in both academia and industry for its numerous applications, such as scene understanding, image retrieval, augmented reality translation (Wu et al. 2019), and robot navigation. Thanks to the surge of deep neural networks, significant progress has been made in detection and recognition separable solutions (Wu and Natarajan 2017; Long et al. 2018; Wang et al. 2019b,a; Zhan and Lu 2019; Shi et al. 2018; Yu et al. 2020; Wan et al. 2019), as well as end-to-end text spotting methods. However, existing end-to-end models (Sun et al. 2018; Liu et al. 2018; Feng et al. 2019) are mostly built on two-stage frameworks or character-based methods (Xing et al. 2019; Lyu et al. 2018) with complex pipelines, which are inefficient for real-time applications. In this paper, we try to investigate a real-time text spotter for arbitrarily-shaped text.
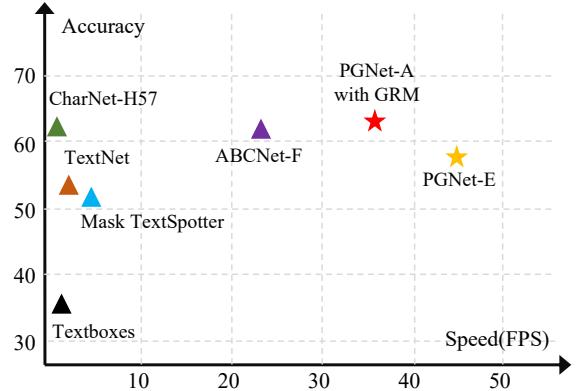
Figure 1: Model Speed vs. Recognition Accuracy on Total-Text: Our PGNet-E achieves at least two times faster than the most recent state-of-the-art method ABCNet (Liu et al. 2020) with competitive recognition accuracy. Complete results are in Table. 3.

Reading of arbitrarily-shaped scene text is a challenging task, as compared in Fig. 2, and the most recent works may suffer from the following disadvantages: (1) The pipelines of two-stage methods (Sun et al. 2018; Lyu et al. 2018; Feng et al. 2019; Liu et al. 2020) are inefficient, which may involve time-consuming Non-maximum Suppression (NMS) and Region of Interest (RoI) operations. Especially for arbitrarily-shaped text spotter, specific RoI transformation operation, such as RoISlide (Feng et al. 2019) or Bezier-Align (Liu et al. 2020), brings non-negligible computational overhead. (2) In Mask TextSpotter (Lyu et al. 2018) and CharNet (Xing et al. 2019), character-level annotations are required for training, which is too expensive to afford. Though CharNet could be trained in a weakly supervised manner by character-level annotations in synthetic datasets, free synthesized data is not completely replaceable for real data in practice. (3) The recognition of text in non-traditional reading directions would be failed with pre-defined rules. For example, TextDragon (Feng et al. 2019) and Mask TextSpotter (Lyu et al. 2018) make a strong assumption that the reading direction of text region is either from left to right or from up to down, which precludes correct recognition of more challenging text.
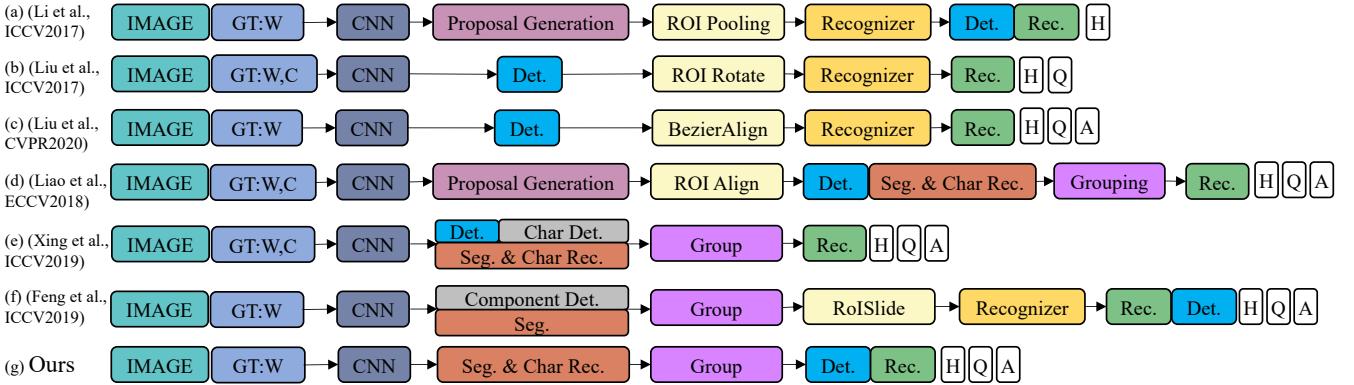
Figure 2: Overview of some end-to-end scene text spotting methods that are most relevant to ours, and the blue and green boxes represent their detection and recognition results. Inside the GT (ground-truth) box, 'W' and 'C' represent word-level and character-level annotation. The 'H', 'Q', and 'A' represent that the method can detect horizontal, quadrilateral, and arbitrarily-shaped text, respectively. Our method is free from character-level annotations, NMS, and RoI operations.

In this paper, we propose a novel framework for reading text in real-time speed with point gathering operation, namely PGNet. The PGNet is a single-shot text spotter based on multi-task learning. The architecture of PGNet is shown in Fig. 3. We employ an FCN (Milletari, Navab, and Ahmadi 2016) model to learn various information of text regions simultaneously, including text center line (TCL), text border offset (TBO), text direction offset (TDO), and text character classification map (TCC). The pixel-level character classification map is trained with a proposed Point Gathering CTC (PG-CTC) loss, making it free from character-level annotations. In the post-processing, we extract the center point sequence in the reading order of each text instance with TCL and TDO maps, and the detection results can be obtained with the corresponding boundary offset information from TBO map. Using the PG-CTC decoder, we serialize the high-level two-dimensional TCC map to character classification probability vector sequences which can be further decoded to the recognition results. The details will be discussed in Sec.3.1. As depicted in Fig. 2 and Fig. 1, our pipeline is simple yet efficient, and experiments on public benchmarks prove that PGNet achieves better or competitive performance with excellent running speed.

Moreover, inspired by SRN (Yu et al. 2020) and GTC (Hu et al. 2020), we propose a graph refinement module (GRM) to make secondary reasoning to improve the end-to-end performance further. The points in a text sequence are formulated as nodes in a graph, where the representation of each node is enhanced with semantic context and visual context information from its neighbors, and the character classification result should be more accurate.

The contributions of this paper are three-fold:

- We propose a simple yet powerful arbitrarily-shaped text spotter (PGNet), which is free from character-level annotations, NMS, and RoI operations, and it achieves better or competitive performance in end-to-end performance with excellent running speed;

- We introduce a mechanism to restore the reading order of

characters in each text instance, making our method able to correctly recognize text in more challenging situations and non-traditional reading directions;

- We also propose an efficient graph refinement module (GRM) to improve the CTC recognition.

2   Related Work

In this section, we will review some representative scene text spotters, as well as some recent progress in graph neural networks. A comprehensive review of recent scene text spotters can be found in (Ye and Doermann 2015; Zhu, Yao, and Bai 2016; Baek et al. 2019; Liao et al. 2021).

Scene Text Spotting. Inspired by the generic object detection methods (Liu et al. 2016; Ren et al. 2015; Redmon et al. 2016) and segmentation methods (He et al. 2017; Milletari, Navab, and Ahmadi 2016), the text spotting methods are developed from spotting regular scene text to spotting arbitrarily-shaped scene text. Lee and Osindero (2016) proposed the first successful end-to-end text recognition model, which only supports horizontal text and requires relatively complex training procedures. To address the multi-orientation problem of text, Bušta et al. (2017) utilize YOLO (Redmon et al. 2016) to generate rotational proposals, and train RoI sampled features with CTC loss. Inspired by the Faster RCNN (Ren et al. 2015), TextNet (Sun et al. 2018) generates text proposals in quadrangles, and encodes the aligned RoI features into context information with a simple recurrent neural network to generate the text sequences, which contains some background information; thus, it may suffer from reading curve texts.

For the spotting of arbitrarily-shaped scene text, Mask TextSpotter (Lyu et al. 2018) detects and recognizes text instances of arbitrary shapes by segmenting the text regions and character regions. However, the character-level annotations are required for training, which is too expensive to afford. Considering the arbitrarily-shaped region of text as a series of quadrangles, TextDragon (Feng et al. 2019) extracts the components of text feature through RoISlide and
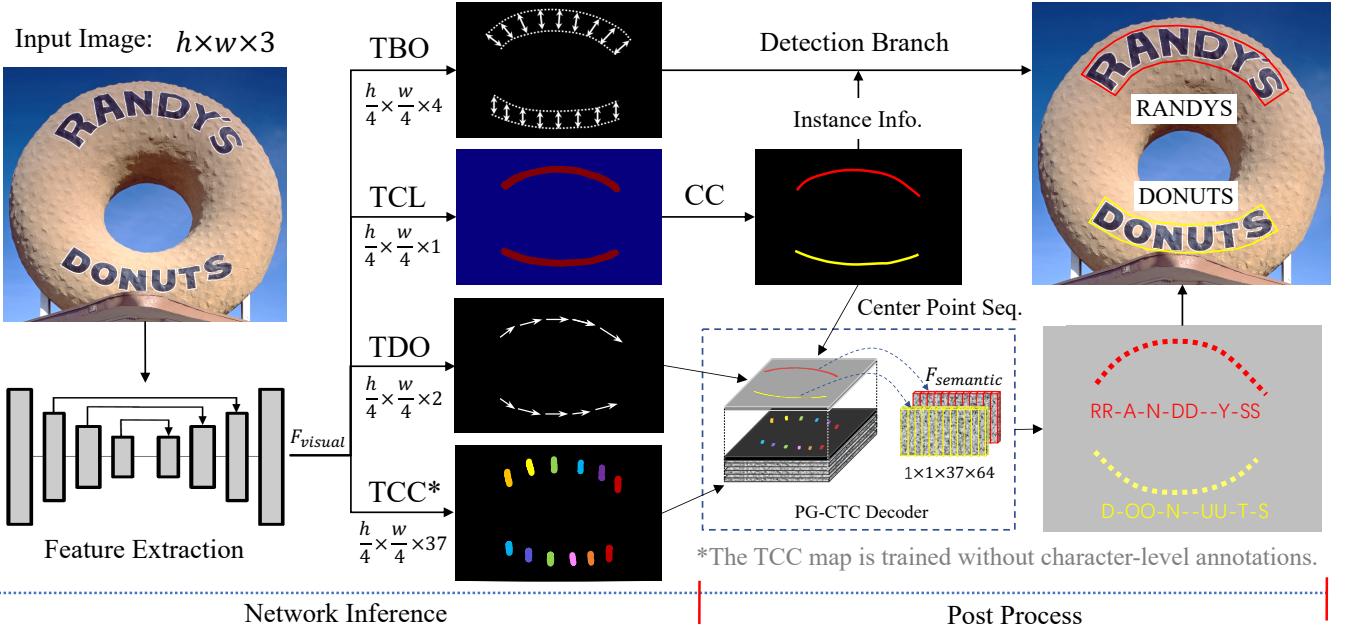
Figure 3: The pipeline of PGNet: 1) Extract feature from an input image, and learn TCL, TBO, TDO, TCC maps as a multi-task problem; 2) The detection and recognition of each text instance can be achieved in a single shot by polygon restoration and PG-CTC decoding mechanism with the center point sequence of each text region.

recognizes each cropped feature with CTC based text recognizer. ABCNet (Liu et al. 2020) attempts to adaptively fit arbitrarily-shaped text by a parameterized Bezier curve to reduce the computation overhead, and they propose a BezierAlign layer for extracting accurate convolution features of a text instance, making it suitable for some real-time scenarios. Mask Textspotter, TextDragon, and ABCNet are capable of spotting arbitrarily-shaped scene text, but they are all RoI-based methods and may involve NMS, RoI cropping, and pooling operations, which is time-consuming and may reduce the performance. CharNet is the first one-stage arbitrarily-shaped scene text spotting method, which requires character-level ground truth data for training and its backbone is too heavy to run in a real-time speed.

Graph Neural Networks. The unstructured data is common in many machine learning tasks and can be organized as graphs. Thanks to the previous effort, we can use the convolutional neural network for graph-structured data. The graph convolutional networks (GCNs) can be categorized into spectral methods and spatial methods. Spectral GCNs generalize convolution by Graph Fourier Transform, while spatial GCNs directly perform manually-defined convolution on graph nodes and their neighbors. For more details about GCNs, please refer to (Wu et al. 2020). In terms of applications, GCNs is proved to be efficient in many computer vision tasks, including scene graph generation, point clouds classification, and action recognition. For example, Zhang et al. (2020) propose a relational reasoning graph network for arbitrary shape text detection by predicting linkages of text components. In this paper, we adopt the Spatial GCN to reasoning the semantic information between point and its neighbors to improve the CTC recognition.

3 Methodology

The architecture of our proposed method is shown in Fig. 3. Firstly, the input image is fed into a stem backbone with FPN to produce feature F_{visual}. Then, the F_{visual} is used to predict TCL, TBO, TDO, and pixel-level TCC map by multi-task learning in parallel at $1/4$ size of the input images. In the training period, the TCL, TBO, and TDO are supervised by the same scale label maps, while a PG-CTC loss is proposed to train the pixel-level TCC map to solve the lack of character-level annotations. In the inference period, we extract the center point sequence of each text instance from TCL, and sort them with TDO information to recover the right reading order, making our method recognize text in non-traditional reading directions correctly. With the aid of the corresponding boundary offset information from TBO, the detection of each text instance can be achieved in a single shot by polygon restoration. Simultaneously, the PG-CTC decoder can serialize high-level two-dimensional TCC map to character classification probability sequences and decode them into final text recognition results.

3.1 Point Gathering CTC

The point gathering (PG) operation plays an important role in both the training and inference process of PGNet and helps to get rid of character-level annotations, NMS, and RoI operations. The TCC maps of PGNet are maps of 37 characters, including 26 letters, 10 Arabic numerals, and one background class. The point gathering operation is employed to gather the character classification probability sequence from the TCC map according to the center points in the center of

each text region, which can be formulated as

$$P_\pi = gather(TCC, \pi), \tag{1}$$

where $\pi = \{p_1, p_2, \ldots, p_N\}$ is a center point sequence of length N, and $p_i = (x_i, y_i)$. The output P_π is the character classification probability sequence with size $N \times 37$.

In the training process, the proposed PG-CTC loss makes the training of pixel-level TCC map free from character-level annotations. The typical CTC loss function tackles the training problem of the source and target sequences with inconsistent lengths with a background class. The CRNN (Shi, Bai, and Yao 2017) framework transforms the height of the feature map to 1, which may suffer from the background noise while recognizing the curved text. The 2D-CTC (Wan et al. 2019) expands the search path of CTC to two-dimensional space, but it still can not handle an image with multi text instances. Here we address the problem by PG-CTC and formulate the classic CTC loss as $CTC\_loss(P, L)$, where P is a character classification probability sequence and L is its transcript label. For an image with M text instances, suppose the center point coordinate sequences are $\{\pi_1, \pi_1, \ldots, \pi_M\}$, and corresponding transcript labels are $\{L_1, L_1, \ldots, L_M\}$, then we define the PG-CTC loss as

$$L_{PG-CTC} = \sum_{i=1}^{M} CTC\_loss(P_{\pi_i}, L_i), \tag{2}$$

where we can calculate the centerline of polygonal word-level annotations and sample it densely to obtain the center point sequences π_i in the training process, instead of using character-level annotations. With the training of big data, the character classification information of each pixel in TCC could be learned.

In the inference process, the PG-CTC decoder dramatically simplifies the overall pipeline of an end-to-end arbitrarily-shaped text spotter, and the NMS and RoI operations are not required in PGNet. For a text region in the TCL map, we extract a center point sequence and sort it in the right reading order, which can be donated as π. Specifically, we adopt a morphological method (Zhang and Suen 1984) to get the skeleton of a text region and treat it as the center point sequence. The text direction of each point can be extracted from TDO maps. We calculate an average direction of all points and sort them according to the length of projection along the direction to obtain the center point sequence π. The character classification probability sequence P_π can be extracted with Eq. (1), and the PG-CTC decoder can be denoted as

$$R_\pi = CTC\_decoder(P_\pi), \tag{3}$$

where R_π represents the transcription of π. For the polygon restoration, we obtain the corresponding border point pairs of π with TBO maps in the same position, and link all the border points clockwise to obtain a complete polygon representation. For more details about polygon restoration, please refer to our previous SAST (Wang et al. 2019a). Compared with the CTC-based CRNN framework, the PG-CTC could handle images with multi-text instances of arbitrary shape, where the application of CTC loss is expanded a lot.
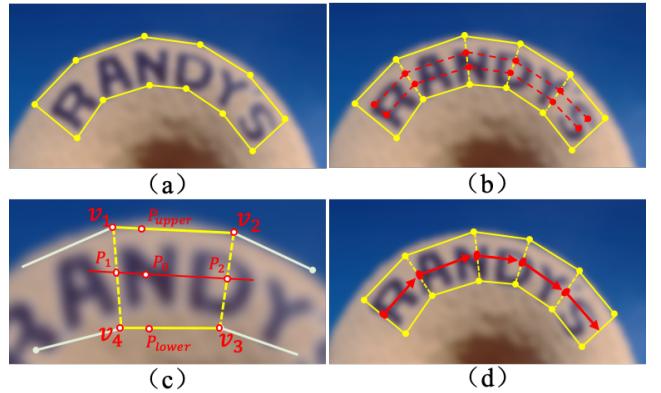

Figure 4: Label Generation: (a) is the ground truth annotation of curved text in yellow; (b)–(d) are the generation of TCL, TBO, and TDO maps, respectively.

3.2 Network Architecture

Considering the limitation of computing resources in different scenarios, we propose two versions of PGNet, that is, PGNet-Accuracy and PGNet-Efficient, denoted by PGNet-A and PGNet-E in the following sections. The only difference is stem network, and the PGNet-A adopts ResNet-50 as the backbone network, while PGNet-E employs EfficientNet-B0. With different levels of feature map from the stem network gradually merged three-times in the FPN manner, a fused feature map F_{visual} is produced at $1/4$ size of the input images. The TCL and the other maps are predicted in parallel, where we adopt a 1×1 convolution layer with the number of output channels set to $\{1, 2, 4, 37\}$ for TCL, TDO, TBO, TCC map, respectively.

3.3 Label Generation

The label generation of arbitrarily-shaped text is shown in Fig. 4. The TCL map is the shrunk version segmentation of the text region. The TBO map indicates the offset between each pixel in TCL and the corresponding point pair in the upper and lower edge of its text region, which helps to determine the boundaries of text regions in the inference. We follow SAST (Wang et al. 2019a) to generate TCL and TBO map, where more details are introduced. Inspired by the reading mechanism of humans that the eye moves from one character to the next character along the centerline of text region while reading, the TDO map is estimated to recover the reading order of scene text components, which benefits both detection and recognition tasks, especially for those scene text in non-traditional reading directions. The TDO map indicates the offset vector of each pixel in the TCL map to the next reading position. For a quadrilateral region annotation, the direction of the offset vector is from the center point of the left edge to the center point of the right edge, and its magnitude is the length of text region normalized by the number of characters. Polygonal annotations of more than four vertices are treated as a series of quadrangles connected together, and TBO map and TDO map can be generated gradually from quadrangles as described before.

3.4 Training Objectives

The loss of multi-task learning can be formulated as

$$L = \lambda_1 L_{tcl} + \lambda_2 L_{tbo} + \lambda_3 L_{tdo} + \lambda_4 L_{tcc}, \quad (4)$$

where L_{tcl}, L_{tbo}, L_{tdo} and L_{tcc} represent the loss of TCL, TBO, TDO and TCC maps. We train TCL branch by minimizing the Dice loss (Milletari, Navab, and Ahmadi 2016), and the Smooth L_1 loss (Girshick 2015) is adopted for TBO and TDO map, while TCC map is trained with PG-CTC loss as mentioned before. The loss weights λ_1, λ_2, λ_3, and λ_4 are set to $\{1.0, 1.0, 1.0, 5.0\}$ empirically.

3.5 Graph Refinement Module

We also propose a graph refinement module to perceive the word-level semantic context and visual context to improve the end-to-end reading performance with GCNs further.

The coarse recognition results is refined in text instance level, and we construct a visual reasoning graph and a semantic reasoning graph for a point sequence π, of which the points are considered as the nodes in a graph. We use the same structure of graph convolution layer as (Wang et al. 2019c) and (Zhang et al. 2020). Especially, the F_{visual} and TCC map are both required as inputs. F_{visual} is the output feature of FPN, as illustrated in Fig.3. For a point sequence $\pi = \{p_1, p_2, \ldots, p_N\}$, the adjacency matrix is defined as

$$\boldsymbol{A_{ij}} = 1 - D(p_i, p_j) / \max(A), \quad (5)$$

where $D(p_i, p_j)$ is a L2 distance between p_i and p_j, and each node is self-connected. The structure of GRM is depicted in Fig.5, and the two numbers in brackets indicate d_{in} and d_{out} of a neural layer. In the semantic reasoning graph, the feature map F_s is obtained by point gathering operation from TCC map and further embedded to X_s of shape $N \times 256$. With three graph layers, the input X_s is transformed to produce Y_s of shape $N \times 64$; In the visual reasoning graph, the features F_v is gathered from F_{visual}, and its channel is transformed from 128 to 256 with several convolutional layers to get X_v. With a similar network, we can get the visual reasoning output Y_v of shape $N \times 64$; Finally, we concatenate Y_v and Y_s, and treat it as a classification problem with several FC layers to produce the refined probability sequence, where the GRM is also optimized with CTC loss. It is worth mentioning that we pad the coarse recognition sequence to the same length and batched together for efficient training, and the max length is set to 64.

4 Experiments

4.1 Datasets

The benchmark datasets used for the experiments in this paper are briefly introduced below.

ICDAR 2015. The ICDAR 2015 dataset (Karatzas et al. 2015) is collected for the ICDAR 2015 Robust Reading Competition, with 1,000 natural images for training and 500 for testing. The text instances are annotated in word-level.

Total-Text. The Total-Text (Ch'ng and Chan 2017) is another curved text benchmark, which consists of 1,255 training images and 300 testing images with multiple orientations: horizontal, multi-Oriented, and curved.

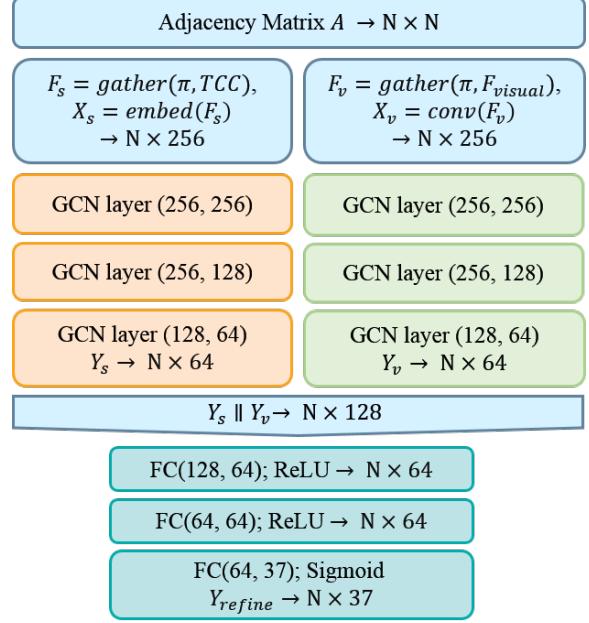input: $\pi = \{p_1, p_2, \ldots, p_N\}$, F_{visual}, TCC map;

Figure 5: The structure of GRM. For each text sequence, we construct a visual graph and a semantic graph, respectively, and their output Y_v and Y_s are concatenated together for further character classification with several FC layers.

4.2 Implementation Details

Training. The stem network is initialized with pre-trained weight on ImageNet (Deng et al. 2009). The training process is mainly divided into the warming-up step, fine-tuning step, and training step of the GRM module. In the warming-up step, we apply Adam optimizer to train our model with learning rate 1e-3, and the learning rate decay factor is 0.94 on the SynthText (Gupta, Vedaldi, and Zisserman 2016); In the fine-tuning step, the learning rate is re-initiated to 1e-3, and the model is tuned on ICDAR2015 and Total-Text data; The GRM is an additional module in our pipeline, and in the last step, we only train the learnable parameters in GRM module to provide both the metrics on ICDAR 2015 and Total-Text with/without GRM for a fair comparison.

The experiments are performed on a workstation with the following configuration, CPU: Intel(R) Xeon(R) CPU E5-2620; GPU: NVIDIA TITAN Xp $\times 4$; RAM: 64GB.

4.3 Ablation Study

In this section, comprehensive experiments are conducted to prove the strengths of the proposed method.

The Effectiveness of TDO. The TDO map is adopted to recover the sequence information of text in non-traditional reading directions, which benefits both detection and recognition tasks. To verify the efficiency of TDO, we conduct several experiments in ICDAR2015 and Total-Text for PGNet-A and PGNet-E. As shown in Tab. 1, the benefits of TDO in Total-Text are apparent, and the recognition accuracy in PGNet-E obtains 1.6% improvement, while it is less

Data Set	ICDAR2015		Total-Text	
Method	PGNet-A	PGNet-E	PGNet-A	PGNet-E
with TDO	62.3	57.4	61.7	58.4
w/o TDO	61.9	56.9	60.4	56.8
Gain	+0.4	+0.5	+1.3	+1.6

Table 1: Evaluation the effectiveness of TDO to improve the recognition performance.

Data Set	ICDAR2015		Total-Text	
Method	PGNet-A	PGNet-E	PGNet-A	PGNet-E
with GRM	63.5	58.7	63.1	60.6
w/o GRM	62.3	57.4	61.7	58.4
Gain	+1.2	+1.3	+1.4	+2.2

Table 2: Evaluation the effectiveness of GRM to improve the recognition performance.

than 0.5% in ICDAR2015. We claim that the non-traditional texts are more likely to appear in the curved Total-Text, while the texts in ICDAR2015 are mostly quadrangles, and the predefined rules may cover most situations.

The Effectiveness of GRM. The GRM is proposed to model word-level semantic context implicitly and improve the end-to-end reading performance. To verify its efficiency, we conduct several experiments in ICDAR2015 and Total-Text on both PGNet-A and PGNet-E, respectively. As depicted in Tab. 2, the efficiency of GRM is proved, and the end-to-end reading performance is improved significantly. Especially, it gains 2.2% for end-to-end recognition tasks for PGNet-E in Total-Text.

4.4 Evaluation for Curved Text

On Total-Text, we evaluate the performance of PGNet for spotting scene text of arbitrary shapes. We fine-tune our model with Total-Text training set and partial ICDAR2019 ArT (Chng et al. 2019), where we remove the images from the Total-Text test set by comparing the MD5 value of each image. The ratio of these two datasets is set to 3: 2 during training. The GRM is further trained with SynthText, partial ICDAR2019 ArT, and Total-Text, of which the data ratio is set to 6: 2: 2, while the learnable parameters outside GRM are frozen. In the inference phase, the longer side of images is resized to 640, with the aspect ratio kept.

The results are shown in Tab. 3 and Fig. 6. Our method achieves the state-of-the-art detection result and is superior to the other competitors by almost 1.5%. Besides, with the help of efficient post-processing, our PGNet-A also achieves comparable end-to-end recognition results. Specifically, compared with the previous most accurate method CharNet H-57, our method has almost the same recognition accuracy, but it is over 30 times faster and free from character-level annotations. Compared with the ABCNet, which is the current fastest arbitrarily-shaped text spotter with a speed of 20+ FPS, our PGNet-E runs nearly two times faster than ABCNet-F with better recognition accuracy. The comparisons with the previous state-of-the-arts demonstrate the efficiency and effectiveness of our method.

4.5 Evaluation for Multi-oriented Text

We also conduct experiments on ICDAR2015 to confirm the superiority of the proposed method on the multi-oriented scene text. Following the same training strategy as Total-Text, The ICDAR2017-MLT-Latin and ICDAR2015 are used in fine-tune stage, of which the data ratio is 3:7, while SynthText, ICDAR2017-MLT-Latin (Nayef et al. 2017) and ICDAR2015 with a ratio of 6:2:2 are adopted for training GRM. In the test phase, the longer side of images is resized to 1536, with the aspect ratio kept.

The results are shown in Tab. 4 and Fig. 6. Our method achieves 88.2% F-Measure on the text detection task and is better than most previous methods. Besides, on the end-to-end recognition task, the PGNet-A with strong and weak lexicon achieves state-of-the-art accuracy. Particularly, our method exceeds the previous best one-stage text spotter CharNet R-50 by 3.2%, 3.8%, and 1.2% when evaluated with strong, weak, and general lexicon, respectively.

4.6 Runtime

In this paper, we propose a simple yet powerful arbitrarily-shaped text spotter, and considering the limitation of computing resources in different application scenarios, we propose PGNet-A and PGNet-E with different stem networks. The runtime of our method can be roughly divided into three parts: network inference stage, post-processing stage, and graph refinement stage. The runtime of PGNet on Total-Text is evaluated with NVIDIA Tesla V100-SXM2-16GB, which is the same as ABCNet. The longer side of the test image is resized to 640, and the batch size is set to 1 on a single GPU. It takes 15.1 ms, 6.4 ms, and 3.2 ms in the three stages, respectively. It is worth noting that the post-processing stage, which accounted for a large proportion of time cost, is executed with Python code and can be further optimized. In Total-Text, PGNet-E runs at 46.7 FPS with 84.8% and 58.4% F-Measure respectively on the detection and end-to-end recognition without lexicon, surpassing the previous arbitrarily-shaped text spotters in efficiency significantly, as depicted in Tab. 3.

5 Conclusion and Future Work

In this paper, we propose a real-time arbitrarily-shaped text spotter PGNet together with a novel graph refinement module to relieve the inefficient problem of previous methods. Optimizing TCC maps with proposed PG-CTC loss, our PGNet is free from character-level annotations and gets rid of NMS and RoI operations in the inference stage which is single-shot, efficient and straightforward. Moreover, a novel graph refinement module is proposed to improve the end-to-end performance further by reasoning the word-level semantic information via a spatial graph neural network. The contributions of our paper are confirmed by comprehensive experiments and qualitative visualization in public benchmarks. In the future, we are interested in promoting the deployment of our novel real-time arbitrarily-shaped text spotter on smart edge devices.

Method	Backbone	Detection			Recognition	FPS
		Recall	Precious	F-score	None	
FOTS (Liu et al. 2018)	ResNet-50	38.0	52.3	44.0	32.2	-
Textboxes (Liao et al. 2017)	ResNet-50-FPN	45.5	62.1	52.5	36.3	1.4
Mask TextSpotter (Lyu et al. 2018)	ResNet-50-FPN	55.0	69.0	61.3	52.9	4.8
TextNet (Sun et al. 2018)	ResNet-50-SAM	59.5	68.2	63.5	54.0	2.7
TextSnake (Long et al. 2018)	VGG16-FPN	74.5	82.7	78.4	-	-
TextDragon (Feng et al. 2019)	VGG16-FPN	75.7	85.6	80.3	48.8	-
CharNet H-57 (Xing et al. 2019)	Hourglass-57	81.0	88.6	84.6	63.6	1.2
Boundary (Wang et al. 2020)	ResNet-50-FPN	85.0	**88.9**	**87.0**	**65.0**	-
ABCNet-F (Liu et al. 2020)	ResNet-50-FPN	-	-	-	61.9	22.8
PGNet-A	ResNet-50-FPN	**86.8**	85.3	86.1	61.7	38.2
PGNet-A with GRM	ResNet-50-FPN	**86.8**	85.5	86.1	63.1	35.5
PGNet-E	EfficientNet-B0-FPN	85.0	84.6	84.8	58.4	**46.7**
PGNet-E with GRM	EfficientNet-B0-FPN	84.9	84.7	84.8	60.5	40.5

Table 3: Evaluation on Total-Text for detecting text lines of arbitrary shapes.

Method	Detection			Method	E2E Recognition		
	R	P	F		S	W	G
EAST (Zhou et al. 2017)	78.3	83.3	80.7	OpenCV 3.0 (Karatzas et al. 2015)	13.8	12.0	8.0
LOMO (Zhang et al. 2019)	81.0	91.6	86.0	Deep TextSpotter (Bušta et al. 2017)	54.0	51.0	47.0
Mask TextSpotter (Lyu et al. 2018)	81.0	91.6	86.0	Mask TextSpotter (Lyu et al. 2018)	79.3	73.0	62.4
TextNet (Sun et al. 2018)	85.4	89.4	87.4	TextNet (Sun et al. 2018)	78.7	74.9	60.5
FOTS R-50 (Liu et al. 2018)	85.2	91.0	88.0	FOTS R-50 (Liu et al. 2018)	81.1	75.9	60.8
TextDragon (Feng et al. 2019)	83.8	**92.5**	87.9	TextDragon (Feng et al. 2019)	82.5	78.3	**65.2**
CharNet R-50 (Xing et al. 2019)	**88.3**	91.2	**89.7**	CharNet R-50 (Xing et al. 2019)	80.1	74.5	62.2
Boundary (Wang et al. 2020)	87.5	89.8	88.6	Boundary (Wang et al. 2020)	79.7	75.2	64.1
TextPerceptron (Qiao et al. 2020)	82.5	92.3	87.1	TextPerceptron (Qiao et al. 2020)	80.5	76.6	65.1
PGNet-A	84.3	92.4	88.1	PGNet-A	82.9	77.7	62.3
PGNet-A with GRM	84.8	91.8	88.2	PGNet-A with GRM	**83.3**	**78.3**	63.5
PGNet-E	83.6	85.6	84.6	PGNet-E	80.2	74.9	57.4
PGNet-E with GRM	83.6	85.8	84.7	PGNet-E with GRM	80.5	75.3	58.7

Table 4: Evaluation on ICDAR 2015 for detecting oriented text. "P", "R", "F" represent "Precision", "Recall", "F-measure" respectively. "S", "W", "G" represent recognition with "Strong", "Weak", "Generic" lexicon respectively.

Figure 6: Qualitative results of our method on Total-Text (left two columns) and ICDAR15 (right two columns) datasets.

6 Appendix

6.1 Graph Convolution Layer

A graph convolution layer takes as input a node feature matrix X together with an adjacency matrix A and outputs a transformed node feature matrix Y. Formally, a graph convolution layer in our case has the following formulation,

$$Y = \sigma([X \| GX]W), \qquad (6)$$

where $X \in \mathbb{R}^{N \times d_{in}}$, $Y \in \mathbb{R}^{N \times d_{out}}$, N is the number of nodes, and d_{in}, d_{out} are the dimension of input/output node features. $G = g(X, A)$ is a symmetric normalized Laplacian matrix of size $N \times N$, and $g(\cdot)$ is a function of X and A. Operator $\|$ represents matrix concatenation along the feature dimension. W is the learnable weight matrix with size $2d_{in} \times d_{out}$, and $\sigma(\cdot)$ is the non-linear activation function. In our paper, mean aggregation is adopted as $g(\cdot)$ for the aggregation operation,

$$G = \Lambda^{-\frac{1}{2}} A \Lambda^{-\frac{1}{2}}, \qquad (7)$$

where Λ is a diagonal matrix with $\Lambda_{ii} = \sum_j A_{ij}$. The topological structure is implicit in the adjacency matrix, and A_{ij} indicates the affinity between the i-th node and j-th node. For a point sequence $\pi = \{p_1, p_2, \ldots, p_N\}$, the adjacency matrix is defined as

$$A_{ij} = 1 - D(p_i, p_j)/\max(A), \qquad (8)$$

where $D(p_i, p_j)$ is a L2 distance between p_i and p_j, and the diagonal elements of the adjacency matrix is always 1, which means that each node is self-connected.

6.2 Visualization and Discussion

We will do some visualization and case analysis to prove the effectiveness of PGNet and GRM qualitatively. In Fig. 6, Fig. 7 and Fig. 8, a green polygon means that detection and recognition both hit the ground truth. A blue polygon with red text indicates that the detection result hits, but the recognition result is incorrect. The right recognition results are not displayed for clear visualization. Besides, the upper left vertex of the text polygon in the right reading order is marked with a pink dot, if it exists.

Qualitative results of PGNet. The visualization of spotting results in ICDAR2015 and Total-Text are shown in Fig. 6. As can be seen, the proposed arbitrarily-shaped text spotter PGNet can handle curved and multi-oriented text regions well. The superiority of TDO is proved in Fig. 8. The reading order information of group A is recovered by predefined rules, which is the same as (Feng et al. 2019). Furthermore, group B is yielded by TDO map. As shown in Fig. 6,
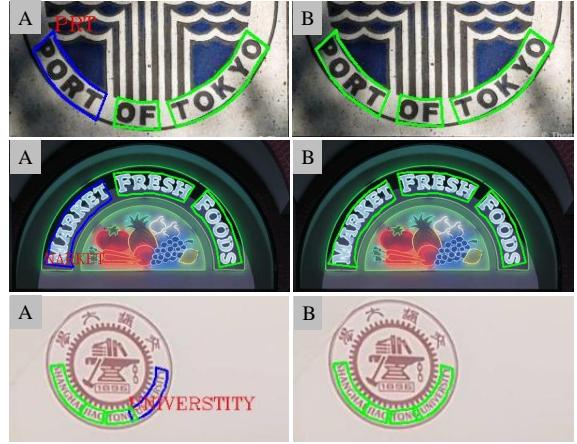

Figure 7: Qualitative results of our GRM to correct error recognition, where group A/B is the end-to-end results without/with GRM.
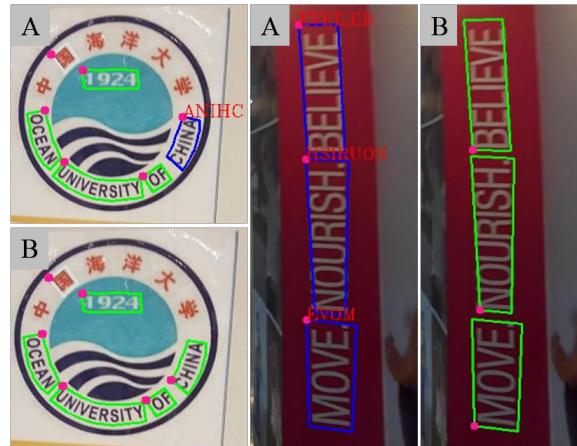

Figure 8: Qualitative results of our TDO to correctly recognize text in nontraditional reading directions, where group B/A is the end-to-end results with/without TDO.

the word "MOVE" is misidentified as "EVOM" with predefined rules, while is recognized correctly with the TDO map.

Qualitative results of GRM. The capability of GRM is depicted in Fig. 7, and group A and group B is the end-to-end results without/with GRM. We show that the GRM can correct the recognition results in different scenarios. In the first case, the wrong recognition "PRT" is revised to the correct "PORT", proving the ability of GRM to make up for missing characters. In the second case, the wrong recognition "NARKET" is refined to the correct "MARKET", indicating the capability of correcting the wrong recognition of characters caused by similar appearance. It is shown in the third case that the ability of GRM to delete extra characters by correcting "UNIVERSTITY" to "UNIVERSITY". The cases mentioned above show that the GRM can add, modify and delete the error characters, which fully prove the potential of the GRM.

References

Baek, J.; Kim, G.; Lee, J.; Park, S.; Han, D.; Yun, S.; Oh, S. J.; and Lee, H. 2019. What is wrong with scene text recognition model comparisons? dataset and model analysis. In *Proceedings of the IEEE International Conference on Computer Vision*, 4715–4723.

Bušta, M.; Neumann, L.; Matas, J.; and xxx. 2017. Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework. In *IEEE Int. Conf. Comp. Vis. (ICCV)*, 2223–2231.

Ch'ng, C. K.; and Chan, C. S. 2017. Total-Text: A comprehensive dataset for scene text detection and recognition. In *Int. Conf. Doc. Anal. Recognit. (ICDAR)*, volume 1, 935–942. IEEE.

Chng, C. K.; Liu, Y.; Sun, Y.; Ng, C. C.; Luo, C.; Ni, Z.; Fang, C.; Zhang, S.; Han, J.; Ding, E.; Liu, J.; Karatzas, D.; Chan, C. S.; and Jin, L. 2019. ICDAR2019 Robust Reading Challenge on Arbitrary-Shaped Text - RRC-ArT. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 1571–1576.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 248–255. IEEE.

Feng, W.; He, W.; Yin, F.; Zhang, X.-Y.; and Liu, C.-L. 2019. TextDragon: An End-to-End Framework for Arbitrary Shaped Text Spotting. In *IEEE Int. Conf. Comp. Vis. (ICCV)*, 9076–9085.

Girshick, R. 2015. Fast R-CNN. In *IEEE Int. Conf. Comp. Vis. (ICCV)*, 1440–1448.

Gupta, A.; Vedaldi, A.; and Zisserman, A. 2016. Synthetic data for text localisation in natural images. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 2315–2324.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask R-CNN. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 2961–2969.

Hu, W.; Cai, X.; Hou, J.; Yi, S.; and Lin, Z. 2020. GTC: Guided training of ctc towards efficient and accurate scene text recognition. *arXiv preprint arXiv:2002.01276* .

Karatzas, D.; Gomez-Bigorda, L.; Nicolaou, A.; Ghosh, S.; Bagdanov, A.; Iwamura, M.; Matas, J.; Neumann, L.; Chandrasekhar, V. R.; Lu, S.; et al. 2015. ICDAR 2015 competition on robust reading. In *Int. Conf. Doc. Anal. Recognit. (ICDAR)*, 1156–1160. IEEE.

Lee, C.-Y.; and Osindero, S. 2016. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2231–2239.

Liao, M.; Lyu, P.; He, M.; Yao, C.; Wu, W.; and Bai, X. 2021. Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43(2): 532–548. doi:10.1109/TPAMI.2019.2937086.

Liao, M.; Shi, B.; Bai, X.; Wang, X.; and Liu, W. 2017. TextBoxes: A fast text detector with a single deep neural network. In *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 4161–4167.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. SSD: Single shot multibox detector. In *Eur. Conf. Comp. Vis. (ECCV)*, 21–37. Springer.

Liu, X.; Liang, D.; Yan, S.; Chen, D.; Qiao, Y.; and Yan, J. 2018. FOTS: Fast oriented text spotting with a unified network. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 5676–5685.

Liu, Y.; Chen, H.; Shen, C.; He, T.; Jin, L.; and Wang, L. 2020. ABCNet: Real-time Scene Text Spotting with Adaptive Bezier-Curve Network. *arXiv preprint arXiv:2002.10200* .

Long, S.; Ruan, J.; Zhang, W.; He, X.; Wu, W.; and Yao, C. 2018. TextSnake: A flexible representation for detecting text of arbitrary shapes. In *Eur. Conf. Comp. Vis. (ECCV)*, 20–36.

Lyu, P.; Liao, M.; Yao, C.; Wu, W.; and Bai, X. 2018. Mask TextSpotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. In *Eur. Conf. Comp. Vis. (ECCV)*, 67–83.

Milletari, F.; Navab, N.; and Ahmadi, S.-A. 2016. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In *4th Int. Conf. 3D Vision (3DV)*, 565–571. IEEE.

Nayef, N.; Yin, F.; Bizid, I.; Choi, H.; Feng, Y.; Karatzas, D.; Luo, Z.; Pal, U.; Rigaud, C.; Chazalon, J.; et al. 2017. ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification-rrc-mlt. In *Int. Conf. Doc. Anal. Recognit. (ICDAR)*, volume 1, 1454–1459. IEEE.

Qiao, L.; Tang, S.; Cheng, Z.; Xu, Y.; Niu, Y.; Pu, S.; and Wu, F. 2020. Text perceptron: Towards end-to-end arbitrary-shaped text spotting. *arXiv preprint arXiv:2002.06820* .

Redmon, J.; Divvala, S.; Girshick, R.; and Farhadi, A. 2016. You Only Look Once: Unified, real-time object detection. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 779–788.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Adv. Neural Inf. Process. Syst. (NIPS)*, 91–99.

Shi, B.; Bai, X.; and Yao, C. 2017. An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 39(11): 2298–2304.

Shi, B.; Yang, M.; Wang, X.; Lyu, P.; Yao, C.; and Bai, X. 2018. Aster: An attentional scene text recognizer with flexible rectification. *IEEE Trans. Pattern Anal. Mach. Intell.* 41(9): 2035–2048.

Sun, Y.; Zhang, C.; Huang, Z.; Liu, J.; Han, J.; and Ding, E. 2018. TextNet: Irregular Text Reading from Images with an End-to-End Trainable Network. In *Asian Conf. Comp. Vis. (ACCV)*, 83–99. Springer.

Wan, Z.; Xie, F.; Liu, Y.; Bai, X.; and Yao, C. 2019. 2D-CTC for Scene Text Recognition. *arXiv preprint. arXiv:1907.09705 [cs.CV]* .

Wang, H.; Lu, P.; Zhang, H.; Yang, M.; Bai, X.; Xu, Y.; He, M.; Wang, Y.; and Liu, W. 2020. All You Need Is Boundary: Toward Arbitrary-Shaped Text Spotting. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(07): 12160–12167. doi:10.1609/aaai.v34i07.6896.

Wang, P.; Zhang, C.; Qi, F.; Huang, Z.; En, M.; Han, J.; Liu, J.; Ding, E.; and Shi, G. 2019a. A Single-Shot Arbitrarily-Shaped Text Detector based on Context Attended Multi-Task Learning. In *Proc. ACM Int. Conf. Multimedia (MM)*, 1277–1285.

Wang, W.; Xie, E.; Li, X.; Hou, W.; Lu, T.; Yu, G.; and Shao, S. 2019b. Shape Robust Text Detection With Progressive Scale Expansion Network. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 9336–9345.

Wang, Z.; Zheng, L.; Li, Y.; and Wang, S. 2019c. Linkage-based Face Clustering via Graph Convolution Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, L.; Zhang, C.; Liu, J.; Han, J.; Liu, J.; Ding, E.; and Bai, X. 2019. Editing text in the wild. In *Proc. ACM Int. Conf. Multimedia (MM)*, 1500–1508.

Wu, Y.; and Natarajan, P. 2017. Self-organized text detection with minimal post-processing via border learning. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 5000–5009.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2020. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* 1–21. doi:10.1109/TNNLS.2020.2978386. ArXiv:1901.00596 [cs.LG].

Xing, L.; Tian, Z.; Huang, W.; and Scott, M. R. 2019. Convolutional character networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 9126–9136.

Ye, Q.; and Doermann, D. 2015. Text detection and recognition in imagery: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* 37(7): 1480–1500.

Yu, D.; Li, X.; Zhang, C.; Han, J.; Liu, J.; and Ding, E. 2020. Towards Accurate Scene Text Recognition With Semantic Reasoning Networks. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 12113–12122.

Zhan, F.; and Lu, S. 2019. ESIR: End-To-End Scene Text Recognition via Iterative Image Rectification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, C.; Liang, B.; Huang, Z.; En, M.; Han, J.; Ding, E.; and Ding, X. 2019. Look More Than Once: An Accurate Detector for Text of Arbitrary Shapes. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*.

Zhang, S.-X.; Zhu, X.; Hou, J.-B.; Liu, C.; Yang, C.; Wang, H.; and Yin, X.-C. 2020. Deep Relational Reasoning Graph Network for Arbitrary Shape Text Detection. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*.

Zhang, T.; and Suen, C. Y. 1984. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM* 27(3): 236–239.

Zhou, X.; Yao, C.; Wen, H.; Wang, Y.; Zhou, S.; He, W.; and Liang, J. 2017. EAST: An efficient and accurate scene text detector. In *IEEE Conf. Comp. Vis. Patt. Recognit. (CVPR)*, 5551–5560.

Zhu, Y.; Yao, C.; and Bai, X. 2016. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science* 10(1): 19–36.